Role Network Model-Based Support for RESTful Business Processes

Xiwei Xu¹, Xin Ye²*, Liming Zhu¹², Qinghua Lu¹⁴
1. NICTA, 13 Garden St, Eveleigh, NSW 2015, Australia
2. School of Computer Science and Engineering, University of New South Wales, NSW 2052, Australia
3. Institute of Information and Decision Technology, Dalian University of Technology, Dalian 116024, P.R.China
4. China University of Petroleum, Qingdao, P.R.China
*Corresponding Author: Postal Address: Institute of Information and Decision Technology, Dalian University of Technology, No.2 Linggong Road, Dalian City, Liaoning Province, P.R.China, 116024. Tel.:+86 411 84707648; Fax: +86 411 84708342

Abstract
The WWW is increasingly used for process-intensive business scenarios that connect services and applications together. Imperative programming and centralised traditional workflow engines lack the flexibility to support such distributed systems. RESTful Business Process (RESTfulBP) is an architectural style that introduces a set of constraints and uses process fragments to enable process coordination and choices at the endpoints to improve system adaptability, especially for human-intensive Web-based business processes. However, the original solution had two limitations: 1) the endpoint process fragments cannot distinguish between obligation and permission semantics to communicate richer choices of next-steps, and 2) it was time-consuming to create process fragment variants in an imperative style (either using complex workflow models or manually). In this paper, we propose a rule-based approach using the Role Network Model (RNM) to address these limitations. We introduce deontic logic into a process fragment and enable declarative generation of process fragments from rules. The proposed solution is evaluated in a real-world property valuation application using the B2B standard process from the Australian lending industry. The evaluation indicates that the new approach has improved both the design and runtime adaptability of the resulting system. Through surveys of other users employing our approach, it further demonstrates improved expressiveness, usability and general usefulness.

Keywords
Workflow management; Business Process; Human-intensive systems; REST; Rule-based; RNM

1. Introduction
The WWW is increasingly used for process-intensive business scenarios that connect services and applications together, often in a distributed manner using the Web infrastructure. The design practices behind the process-intensive applications have evolved from early ad-hoc construction to the recent Web Service approaches using XML-based standards, such as SOAP, WSDL (Web Service Definition Language) and BPEL (Business Process Execution Language). This style of Web Service arose from the history of Remote Procedure Call (RPC) [1] programming.

From a software architecture point of view, the imperative programming paradigm adopted by BPEL and many other flow-based business process languages is not aligned with the Web's declarative style [2, 3]. The declarative style describes what the programme should accomplish, while the imperative style describes how it is accomplished. The architectural mismatch between the underlying data-centric
declarative style of the Web infrastructure and the upper-layer operation-centric imperative style of the Web Service approaches hinders reuse of the mechanisms provided by the Web infrastructure [4]. Industry has also proposed the concept of “subjunctive programming” [5], which advocates Web-based programming by thinking “what if” rather than “if then”. The latter is often represented by a centralised definition of a business process using a language, such as BPEL.

BPEL and many other business process languages are designed to automate the execution and management of business processes. However, in human-intensive collaborative processes, process workers do not sequence their activities in the manner of a procedural software programme [6]. The workers take actions in response to changes in the states of the resources they can access, the knowledge and experience they have, and interactions with other process workers. Although BPEL has an extension [7] to support human-intensive tasks, BPEL lacks the mechanisms within an individual service to inform the invocation sequences in relation to other services [8]. It is hard for BPEL to pre-define all possible control-flow paths of a flexible human-intensive process and automate decision making at every split.

An example of a “blind surgeon metaphor” is presented in [9] to illustrate that both operational flexibility and well-defined procedures are needed in a hospital. The medical protocol typically specifies what should be done instead of what can be done. Such descriptions are useful to guide workers. However, it is clear that restricting the workers to the workflow specified in the medical protocol would lead to absurd situations. Traditional workflow process models cannot completely represent the knowledge of the experts and all situations that might occur in general.

Another example is the Administrative Permit System. The applications for the registration of an enterprise might involve tens of administrative permits handled by the corresponding government departments, and the administrative permits are different for different enterprises’ registrations. These possible administration permits may need to be applied by governmental organisations, namely, rules, according to relevant laws, regulations and the enterprises’ features, such as type, business scope and geographical location. The applications, approval and examining activities of the administrative permits that satisfy rules should be performed at run time. However, in a practical workflow management system, those rules are always incomplete or cannot be satisfied with all scenarios. Therefore, in this type of situation, the application and approval and examining activities of some administrative permits are permitted to be done, namely, can be done, and should be determined whether to be done by governmental officers at run time.

The third example is a more common recruitment process in companies. A company’s recruitment process is highly flexible and can vary according to the position or circumstances. Normally, the technical interview could be performed through an on-line test or written exam, which is optional, especially for senior candidates. The face-to-face interview, on the other hand, is required by the company and must be performed. Depending on factors such as the number and quality of the candidates, the company might select a suitable way to interview the candidate. Using traditional process model language, many process models may need to be created to cover all possible scenarios.

Our previous work, RESTfulBP [4], provides a declarative method to design and execute Web-based business processes that aligns with the Web’s style - REST (REpresentational State Transfer) [10]. In RESTfulBP, processes are described using the notion of a set of process fragments. Each process fragment at an endpoint describes all the possible next-steps or sub-processes with some meta-information about the steps, such as access points and basic control flow semantics. Importantly, the decentralised process fragments are defined inside individual task entities (rather than in a central location) at design time. The process fragments can be exchanged at runtime among process participants to dynamically coordinate their actions. At runtime, a person at an endpoint might be required to select the appropriate next-steps among all the allowed next-steps within a process fragment on the basis of the process status, the execution environment and the person’s knowledge and experience [9]. The selected step(s) is executed by the RESTfulBP process engine either at the server side or inside one’s browser [11]. As shown in
Figure 1, when the RESTfulBP process engine completes the execution of a process fragment, the process instance can suspend and provide a set of new possible process fragments (containing routing information) to the decision maker at an endpoint. The process instance proceeds again once the human decides amongst these process fragments. RESTfulBP can improve the adaptability of the produced systems at both design time and runtime. However, the possible next-steps do not contain deontic information such as what should be done and what can be done. This significantly hinders knowledge workers. This is also one of the common problems of contemporary workflow management systems [9]. In addition, the current RESTfulBP can generate process fragments only from a basic process model. The process fragments for more complex process variants and advanced control-flow patterns are created manually or require an extremely complex workflow model capturing all possibilities.

![Figure 1. Execution of RESTful business processes](image)

This paper introduces rule-based support into RESTfulBP using the Role Network Model (RNM) [12-15] to solve the two limitations mentioned above. RNM is a method for modelling role-based business rules. Compared to traditional input-task-output process techniques, such as BPEL, the techniques based on role theory have the advantage of supporting the principle of separation of duties (SoD) [16]. RNM can describe the behaviours of actors using process-related business rules. In addition, its deontic logic operator can distinguish the tasks that should be done and the tasks that can be done by the corresponding actors with different roles [12, 13]. The rule-based support proposed in this paper is largely targeted at the process-fragment level, although it can be used at the process-modelling level. Note that the main purpose of our approach is not to invent yet another process modelling notation, but to integrate deontic support at the design and implementation levels through process fragments that are directly used by the systems. At the process modelling level, the rule-based process definition is introduced as a complement to BPMN modelling, which can reduce the complexity of variant modelling and facilitate process fragment generation. At the process-fragment level, deontic logic operators are introduced to indicate obligations and permissions within a process fragment for certain role-based actors. This paper is organised as follows: Section 2 discusses the related work. Section 3 provides the background of RESTfulBP and RNM. Section 4 discusses the details of RNM-based RESTfulBP, and the architecture of RNM based RESTfulBP is presented in Section 5. Section 6 evaluates the proposed approach through a real-world property valuation application using the B2B standard process from the Australian lending industry. Section 7 concludes our work and discusses the future work.

## 2. Related Work

To support business process systems better using the basic Web, proposals for REST-based design of business process systems have emerged from both industry and research communities in recent years. In industry, the HTTP protocol, especially using Hypermedia As The Engine Of Application State (HATEOAS), has been used to build workflow systems [5, 17]. HATEOAS has a number of limitations, including a lack of information that causes ambiguous and limited support of classical workflow patterns [4], and limited exploitation of HTTP mechanisms. Compared to RESTfulBP, HATEOAS also does not have the notion of process fragments, which are more complex structures beyond mere next-steps and could be transferred and executed locally. In the research community, RESTful APIs were proposed for workflow management systems such as [18]. Other efforts from the Web Service and process management domain include the
description languages and modelling of RESTful Web services and web resources [2, 19-21] and the extensions of process specification languages that support RESTful interfaces [22-24]. These RESTful approaches for business processes focus on building a RESTful facade to existing workflow or service technologies without fundamentally using the REST principles for the design of business process systems [4]. Our earlier work, RESTfulBP [4, 8, 25], proposes an architectural style that fundamentally aligns the REST principles with the Web-based business processes, as well as a systematic approach for design and execution of Web-based business processes. However, the current solution still has limitations on communicating richer deontic semantics, representing all advanced workflow patterns [26] and automated generation of process fragments. This paper combines RESTfulBP with RNM to overcome these limitations.

In the case-handling design paradigm [9, 65, 66], all information related to a specific case (or process instance) is explicitly provided to avoid context tunnelling. Processes are driven on the basis of the information available rather than the activities already executed. Unlike traditional workflow management, which uses predefined process control structures to determine what should be performed during a business process, case-handling focuses on what can be done to achieve a business goal. The concept of knowledge worker is introduced; this worker is the person in charge of a particular case and makes decisions on how the goals of the corresponding case are reached. Case handling can usually support more flexible and knowledge-intensive business processes. However, it is not clear how more flexible process choices are communicated and described to which knowledge workers. In our approach, the person responsible for making decisions can be regarded as the knowledge worker. Choices about processes are communicated and presented to these knowledge workers at runtime via process fragments with deontic logic.

Business rules are statements to influence and guide the behaviour and information in an organisation [27]. These statements can capture and implement business policies and practices. Business rules change quite often. One problem with traditional business process systems is that they are too rigid to cope with changing business rules [28]. Thus, many approaches have been proposed to model business processes using business rules [29-33]. Deontic logic has been applied to business rules to describe business policy more accurately and comprehensively and to standardise the behaviour of relatively autonomous participants in business processes. Three deontic logic operators of ‘permitted’, ‘prohibited’ and ‘obliged’ have been used [34-37]. Further, some studies suggest that some modifications/extensions of the original form of deontic logic for business process modelling are used instead of permitted [38, 39], such as recommended and anti-recommended. In the service-oriented architecture (SOA) community, hybrid approaches have also been proposed to integrate process-based and rule-based approaches to support more flexible and adaptable business processes at runtime. Most of the complementary rule-based approaches are based on BPEL. These approaches include introducing rule-based engines [40-43], providing rule services [28], and using aspect-oriented programming techniques [44]. The missing link is how rules are used inside runtime artefacts in a distributed manner at endpoints rather than a centralised engine. The RNM based rule approach was selected because it fits better with the decentralised process definition in which the process fragments are divided largely based on the separation of duties of knowledge workers.

Role theories are about the patterns of human conduct, the context and social structure and the individual response. Role-based techniques have also been introduced into business process modelling and design [6, 45, 46]. Role-based approaches can reflect the structure of an organisation, highlight the responsibilities of different actors and improve the understanding of how the responsibilities are achieved. There are several fundamental concepts that support role-based techniques. The first is Business Object, which is an object-oriented representation of the concept of interest in organisation, such as activity and resource [45]. The second is Role, which is a semantic construct of an actor to restrict and define the behaviour of the actor, which could be a human, organisation or agent. Roles can help to meet business process flexibility requirements, especially organisational, functional and operational
requirements because the Role can use Business Rules to embody authority and responsibility of actors. The third concept is Business Rule, which is used to model the collaboration of the business objects in terms of actor and operations by the actor [45] that are allowed. The Business Rules can be formulated based on the Roles [46]. The Business Objects and Roles fit naturally with the Resource-centric view of the RESTfulBP style. Our approach further integrates Business Rules at both the modelling level and the process fragment level through deontic logic and process fragment generation.

3. Background

3.1. RESTfulBP

RESTfulBP [4] is a set of architectural constraints that follow REST principles [10], which restrict the design and execution of Web-based business processes. There are five constraints in the RESTfulBP architectural style: 1) Identify process-entity by URIs; 2) Manipulate process-entity through uniform methods; 3) Expose process state by representation; 4) Communicate process context information by hypermedia; 5) Communicate exception through message headers. RESTfulBP aims to establish communication and coordination mechanisms among participants in a business process from a peer-to-peer and distributed point of view (rather than using centralised coordination supported by traditional workflow engines). Figure 2 shows the concepts of RESTfulBP. Specifically, RESTfulBP exposes the process-entities as URI addressable resources. The task can be implemented through legacy application, Web service or human-intensive service. A resource-oriented “layer” is added to existing services and captures the additional REST and ROA (Resource-oriented Architecture) [47] semantics. All the resources are manipulated through a set of uniform operations. The process-entity resources can provide different kinds of representations to authorized process participants. Process-entities are connected to each other via hyperlinked representations. Routing information is described and exposed at design time in a declarative way by using process fragments. Process fragments are primitive, reusable workflow patterns, which indicate possible next-steps or sub-processes. Process fragments are exchangeable at runtime among process participants to dynamically coordinate their actions. The process fragments enable the declarative description of the process and provide a flexible mechanism for the design and exchange of routing information. The details of RESTfulBP can be found in our earlier work [4, 8]. A design approach [25] was also proposed to give a guideline of creating a business process system that satisfies all the constraints of RESTfulBP. It is proven through systematic case studies that RESTfulBP [48] can have significant benefits in improving the process adaptability at both design time and runtime in real-world situations, comparing real-world situations to traditional business process approaches such as BPEL. RESTfulBP has been integrated with a decision making middleware (MiniZnMASC) to enable automatic selection of process fragments for different classes of user at runtime [49].

![Figure 2. RESTfulBP](image-url)
The routing information in RESTfulBP is described in a declarative way by using process fragments, which are exchanged at runtime among process participants to dynamically coordinate their actions. Based on a set of possible process fragments, the knowledge worker can make a local decision on how to proceed the process instance rather than relying on a centralised process. The process fragments are represented through a specific format based on XLink (XML Linking Language) [50]. The process fragments in RESTfulBP describe the relationship of two related steps, while most of the flow-based process specification languages provide an entire picture of the control flow. The code snippet below is an example of the process fragment that belongs to task 1 in Figure 3.

```
1. <routing xmlns="urn:restful-business-process" xmlns:xlink="http://www.w3.org/1999/xlink">
2.   <nextsteps para="1">
3.     <task xlink:type="simple" xlink:title="task3" xlink:href="/restfulbp/process/task3"
4.         xlink:role="process/roleoftask3" method="PUT"/>
5.     <task xlink:type="simple" xlink:title="task4" xlink:href="/restfulbp/process/task4"
6.         xlink:role="process/roleoftask4" method="PUT"/>
7.   </nextsteps>
8. </routing>
```

Generally, the format of a process fragment is based on the “simple link” [50] of XLink with specific attribute annotations and extra constructs to represent various workflow patterns [51]. The `<nextsteps>` element defines all the executing tasks that need to be synchronized after the execution of the task, while the `<synchronization>` element defines all the executing tasks that need to be executed after the execution of the task. The attribute “para” of the two elements indicate the Branching workflow patterns and Synchronization patterns separately [51]. For example, the value “1” of “para” within `<nextstep>` indicates that the pattern between the current task and the next task is “Sequence”. The value “1” of “para” within `<synchronization>` indicates the pattern between the current task and the previous task is “Sequence”. The task (line 3-4; line 5-6; line 9-10) is described by an XLink. The value “simple” of the attribute “xlink:type” creates a unidirectional hyperlink from the task resource containing this routing information to another task resource. “xlink:title” attribute defines the name of the linked task. “xlink:href” attribute gives the URI of the linked task. “xlink:role” attribute constrains the role of the client that is allowed to execute the task. The value of the attribute is the URI of a specific role resource. The “method” attribute defines the HTTP method required for the task execution.

However, the current RESTfulBP does not provide any mechanism to constrain the behaviours of a knowledge worker. A knowledge worker cannot distinguish between tasks that should be done and tasks that can be done [9] through the routing information contained in a process fragment. In addition, although the current RESTfulBP can automatically generate some process fragments from a corresponding BPMN-based process model, some process fragments have to be created manually by a developer for more complex process variants especially ones that demand advanced workflow patterns [52].
3.2. **Role Network Model (RNM)**

Knowledge workers are the behaviour subjects who have the ability to make decisions in knowledge-intensive systems. The business processes in a knowledge-intensive system, which reflect the behaviour of knowledge workers, are more complicated and flexible than those in industrial automation systems. A knowledge-intensive system can be modelled as a network composed of roles and their coordination relationships. We call our approach Role Network Model (RNM) [12-15]. The main elements of RNM are described as follows. The corresponding meta-model of RNM is shown in Figure 4.

![Figure 4. Meta-model of RNM](image)

1) **Definition of Actor**: all concrete persons and agents are defined as actors. An actor can play different roles.

2) **Definition of Role**: Roles are a set of abstract standards, descriptions, criteria or concepts of an organisation or position in a specific business environment. A role is defined as a nine-tuple: \((N, TYP, ORG, SUP, BOL, BOC, OPERL, OPERC, RES)\). Here, \(N\) represents the name of the role, \(TYP\) represents the type of the role, \(ORG\) represents the affiliated organisation of the role, \(SUP\) represents the superior role, \(BOL\) represents the business object set that can be enquired and handled by the role, \(BOC\) represents the authority set of the role dealing with attributes of business objects, \(OPERL\) represents the operation set of business objects that can be performed by the role \((OPERC\) represents the execution rule set of operations executed by a role, which can describe business-process-related business rules and reflect the behaviour of the corresponding knowledge worker), and \(RES\) represents the resource set assigned to the role to execute an operation.

3) **Definition of Business Object**: The business object is an seven-tuple: \((N, D, P, O, S, IS, FS)\), where, \(N\) represents the name of the business object, \(D\) represents the description of the business object, \(P\) represents the attribute set of the business object, \(O\) represents the operation set of the business object, \(S\) represents the current state of the business object, \(IS\) represents the initial state of the business object, and \(FS\) represents the final state of the business object.
4) **Definition of RoleNetwork**: The role network can be defined based on the main elements mentioned above: RN=(ROL, BO, U, TIM). Where, ROL=\{rol_1, rol_2, ..., rol_m\} represents the set of roles, BO=\{bo_1, bo_2, ..., bo_n\} represents the set of business objects, U=\{u_1, u_2, ..., u_k\} represents the set of actors, and TIM represents the set of discrete time points. RNM emphasises that a role is intelligent, which means it has the capability of making decisions, so that it is more convenient for modelling knowledge-intensive systems. The operating result of a role on business objects can directly affect the subsequent business process. Thus, RNM can better support a dynamic business process.

5) **Definition of ExecutionRule**: The essence of a business process is the coordination of roles played by actors based on OPERC, which is a type of set of business rules. There are several different classification schemas for business rules. OPERC focuses on the statements that are concerned with the invocation of actions in response to events and conditions for a role. These types of rules are called action enablers [53] or reaction rules [54]; they are used to describe the control-flow of the business processes [44]. Every element of OPERC is described by the improved Norm Analysis Method (NAM). NAM [34] reflects the actor’s behaviour rules that standardise the cooperative work among actors. However, NAM lacks a more abstract semantic expression for the actor, and lacks the unified encapsulation and abstract for business information and business operation (function). NAM is extended by replacing the actor with the role, introducing a business object to encase business information and business operation and introducing the time constraint. The improved NAM is expressed as follows.

```plaintext
whenever <condition>
    if <state>
        then <a role>
            is "permitted"/"prohibited"/"obliged"
            to do <BusinessObject.Operation>
            in <Time>
```

In this NAM, <condition> represents the matching context where the rule is to be applied. The context may be the task state business object attributes or the state after their change. The optional <state> represents further conditions that are needed when the roles carry out some operations in this context. "permitted", "prohibited", "obliged" are the three basic operators of deontic logic. Time is a time interval, such as "three hours", and in <Time> means the roles should carry out the operation within the constrained period. This item is optional also. For a particular role and deferent instances of a particular business, business operations executed by the role may be different because of different contexts. Thus, it possibly forms a dynamic business process. And RNM based on the improved NAM can flexibly describe this type of dynamic behaviour of knowledge workers.

6) **Definition of Authority**: The authority represents the attributes of the business object that can be accessed by the role, and the corresponding access permission, such as read and write.

### 4. RNM-Based RESTful Business Process

In this section, we firstly discuss a job recruitment process we use to illustrate our approach as a running example. To integrate RNM into RESTfulBP, we introduce several RNM related process-entities that are discussed in section 4.2. The introduced resources are manipulated through the same set of uniform methods. The corresponding representations of these new process-entities are discussed in section 4.3. Section 4.4 extends the format of context information to support the deontic logic operator that restricts the behaviours of roles.

#### 4.1. Job Recruitment Process

The job recruitment process of a company is a long-running process that requires adaptability to accommodate various scenarios and exceptions. The process also requires interactions among different
roles, including recruiters, candidates, interviewers, managers and administrators, forming a complex process with a mixture of human-intensive and automated tasks. This is very typical for real-world business processes, and motivates our selection of the job recruitment example. Figure 5 gives a simplified job recruitment process as a state chart to emphasise that the details of what and how tasks are performed could be ever-changing and dynamic at runtime. The process fragments (as indicated) can change without affecting the state chart.

The recruiter is the knowledge worker in this process who coordinates the process according to the quality of the candidate, the requirements of the position and other factors. One interesting scenario in this process is organising an interview as mentioned in the introduction section. Normally, the technical interview could be performed through an on-line test or written exam, which is optional, especially for senior candidates. The face-to-face interview, conversely, is required by the company and must be performed. Depending on factors such as the number and the quality of the candidates, the recruiter might select a suitable way to interview the candidate. In the following sections, we will illustrate how we use our approach to express this specific scenario.

![Figure 5. Simplified job recruitment process](image)

### 4.2. RNM Related Process-Entities as Resources

Normally, the business process is modelled by a business manager using an existing modelling notation, such as BPMN [55]. The purpose of process modelling is to identify all the tasks at different levels of abstraction, parties and control dependencies (i.e., sequencing, parallelism) between these tasks. A business process has the following process-entity concepts: process, case (process instance), task, state and routing [56]. In RESTfulBP, the first four types of process-entities are modelled as a set of resources identified by declarative URIs [47], while routing is modelled as a representation of task resources [4, 8]. In addition to the five process-entities, the process modelling of a role-based approach also needs to identify business objects and roles, clarify the mapping between the identified business objects and roles, and describe some of the control dependencies in terms of business rules rather than the notations on the diagram. RNM-based RESTfulBP introduces three new process-entities, namely, Business Object, Role and Rule-based Task.

Business Object is one of the theoretical core concepts in RNM. The OMG Business Object Management Special Interest Group (BOMSIG) has standardised the definition of a business object [57]. A business object is a representation of a thing active in the business domain, including at least its business name, definition, attributes, behaviour, relationships and constraints. Business object is an abstract concept of a specific business domain. The business object in our job recruitment process is the job application. The representation of a business object might be in a natural language, a modelling language, or a programming language. Business objects package business procedures, policies and controls around business data. The business objects are critical for the process performers to perform their works. Business processes can be viewed as the value evolution of business objects through state transitions. A business object is modelled as a resource because it is important enough to be referenced as a thing in itself [47]. The state of a specific business object is represented by the attributes of the business object.
The URI of the business object is the ID of the business object, such as "www.restfulbp.com/recruitment/bo/jobapplication"

Role is another theoretical core concept in RNM. The concept of role contains a set of behaviours, rights and obligations of knowledge workers. RESTfulBP also has the concept of role. However, role in RESTfulBP is an attribute of task resource, which is used to authorise the requester who asks for executing the task. In RNM, the business rules are formulated on the basis of roles. The role concept is explicitly exposed as a separate resource. The role resources are the centralised repository storing the business rules used by the business rule engine. As mentioned earlier, we identify five roles in our job recruitment process, including recruiters, candidates, interviewers, managers and administrators.

Rule-based Task is a new type of task resource that includes business rules as part of its routing information. At the modelling level, the rule-based task resource can be represented by the "business rule task" introduced in BPMN 2.0, which is indicated by a table symbol on the left top of a task activity notation (Rule-based Task). The main difference between a normal task and a rule-based task is that the routing information of the former can be fully extracted from the process model, while the latter uses business rules to describe its routing information. In our solution, the business rules are annotated on the BPMN model. The code generation engine can extract and re-organise the business rules according to the identified roles and include them in the role resources. For example, in our recruitment process, interview is a rule-based task that stores a set of business rules that define what type of interviews should be organised for the candidate.

4.3. RNM-Related Representations

In a RESTful system, the resources are manipulated by referring to their representations. The representation can be any useful information about the state of a resource [47]. In RESTfulBP, a process-entity has multiple representations that both provide different routings and process context information, according to the role of the participant sending the request. In this section, the representations of the RNM-related resources are discussed separately.

4.3.1. Representation of Business Object Resource

The code snippet below shows the structure of the representation of a specific business object. The state of the business object is used to match the condition of the operations in RNM. The state of the business object at a given time is represented by a set of state variables. The <operationset> element includes all the allowed operations of such business object. However, only the link to the operation that can be executed in the given state is included in the representation according to the role–based actor.

```
bo

<table>
<thead>
<tr>
<th>state</th>
<th>ID</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>variable</td>
<td>value</td>
<td>name</td>
</tr>
</tbody>
</table>
```

![Diagram](image)

**Figure 6. Representation of business object resource**

The business object resource has two types of sub-resources. One represents a specific state variable of the business object; the other represents the operation set. The URI of the state variable sub-resource follows the URI template "www.restfulbp.com/recruitment/{caseid}/bo/jobapplication/variables/"
4.3.2. Representation of Role Resource

As the repository of business rules, a role resource stores the business rules that are related to a specific role. The rules in the repository are extracted from the rule-based tasks by the code generation engine at design time. However, the rules still can be manipulated through the representation of the role resource at run-time: for example, adding a rule through PUT method, or deleting a rule through DELETE method. The rules are represented in XML format. Figure 7 shows the structure of a set of rules that belong to a particular role.

![Figure 7. Representation of role resource](image)

The value of the <condition> element of the rule could be the execution state of a certain task or a given process context. Additionally, the values of the <operation> element should be tasks in terms of their corresponding URIs.

4.3.3. Representation of Rule-based Task Resource

The representation of a rule-based task resource includes the business rules that constrain the execution of the task. Because the business rules are not stored as a resource of a rule-based task at runtime, when a rule-based task is requested, the RESTfulBP process engine queries the rule repository to retrieve the related rules of the task and dynamically inserts the business rules into the representation of the rule-based task resource. An <RNM> element is introduced into the representation to store the related rules. The <RNM> element includes an <OPERL> element, which has the same structure as in the representation of a role resource, except a <role> element is added to every <rule> element to indicate the rule’s holder.

4.4. Extension of Context Information

In RESTfulBP, the representation of a case entity has two parts, namely, the business data payload, such as the dates of purchase orders or invoices, and the metadata about the processing context for the payload, which includes the status of the business data and the routing information for the particular case. In RNM-based RESTfulBP, the business data payload and the status of the business data are organised on the basis of business objects. Further, deontic logic is introduced to differentiate the obligation and permission in the routing information. Two elements, <obligation> and <permission>, are introduced into the <nextsteps> element (as discussed in Section 3.1) to represent the deontic logic
Thus, the tasks inside <nextsteps> elements are divided into two groups, one describing what should be done, and the other describing what can be done. A separate element <para> is used to indicate the branching of the obligation. The decision making among the tasks inside the <permission> element is based on the runtime environment and the personal experience of the knowledge worker. In our job recruitment process, for example, the routing information is as below.

```xml
<routing xmlns="urn:restful-business-process" xmlns:xlink="http://www.w3.org/1999/xlink">
    <nextsteps>
        <obligation>
            <para>1</para>
            <task xlink:type="simple" xlink:title="face2face" xlink:href="/www.restfulbp.com/recruitment(face2face)" xlink:role="recruitment/recruitor" method="PUT"/>
        </obligation>
        <permission>
        </permission>
    </nextsteps>
</routing>
```

5. Architecture of the RNM-Based RESTful Business Process

Figure 8 provides the overall architecture of our approach. The dashed line separates the components at design time and runtime. The separation of design time and runtime is from the perspective of the business process produced. At top right is the RESTfulBP modelling tool, which includes the RESTfulBP annotation component and a code generation engine. Code generation is able to generate Task resources (in process fragments) and Rule resources according to the process model and business rules defined by business analysts. At runtime, the RESTfulBP process engine executes the resulting business process using the generated task resources with process fragments and role resources. The process fragments to be executed are selected by the knowledge worker on the basis of his or her personal experience and domain-specific knowledge.

![Figure 8. Architecture of rule-based support for RESTfulBP](image)

5.1. RESTfulBP Modelling Tool

As noted earlier, the main purpose of our approach is not to invent yet another process modelling notation but to integrate deontic support through process fragments that are directly used by the systems. The modelling tool provides an easy way to generate process fragments from less complex BPMN models using a declarative method.
5.1.1. **RESTfulBP annotation plug-in**

The annotation plug-in is based on an Eclipse project called BPMN Modeller [58] that was initially designed by Intalio. BPMN modeller is a business process diagram editor that realises the BPMN (Business Process Modelling Notation) specification [55]. RESTfulBP annotation is a plug-in for the BPMN modeller, which extends several extension points provided by both the BPMN modeller and Eclipse platform (as shown in Table 1).

<table>
<thead>
<tr>
<th>Extension point ID</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>org.eclipse.stp.bpmn.diagram.EAnnotationDecorator</td>
<td>Adding GUI representation to the annotation</td>
</tr>
<tr>
<td>org.eclipse.ui.views.properties.tabbed.propertySections</td>
<td>Adding new property tab</td>
</tr>
<tr>
<td>org.eclipse.ui.views.properties.tabbed.propertyTabs</td>
<td>Adding content to the property tab</td>
</tr>
<tr>
<td>org.eclipse.ui.views</td>
<td>Adding new Eclipse view</td>
</tr>
<tr>
<td>org.eclipse.ui.perspectives</td>
<td>Adding new eclipse perspective</td>
</tr>
<tr>
<td>org.eclipse.ui.actionSets</td>
<td>Adding new Action to the eclipse</td>
</tr>
</tbody>
</table>

The annotation plug-in allows the developer to annotate various elements of the BPMN diagram with RESTful information that is essential for the code generation component to generate the business process systems following REST principles. The RNM-based business rules can be added to the business process models through the rule-based task activity type introduced by BPMN 2.0. However, the current version of the BPMN modeller only supports the BPMN 1.1 specification. Thus, the rules are now added using the annotation mechanism of BPMN. The process model and the corresponding business rules can be defined, designed and later changed by business analysts or process owners.

The data of the business process diagram created by the BPMN Modeler is captured by two xml files. Java memory models are built on top of the two xml files when the editor is active. The Java memory models are highly structured, and can be queried via OCL (Object Constraint Language).

Our modelling tool reorganises the tasks into a tree construct. All tasks are the children of the process node, which is the root of the tree. A task node can contain an arbitrary number of process fragments as its children. It is allowed to manually add more process fragments to a particular task. Figure 9 shows a screenshot of the modelling tool.

![Figure 9. Screenshot of the modelling tool](image)
5.1.2. Code Generation Engine

The code generation engine can extract useful information from the process model, such as the workflow pattern to which a certain task belongs. The engine refers to its pre-and-post tasks and various annotations to generate task resources, with a set of process fragments and role resources that represent the basic business process with business rules modelled by the annotation plug-ins. The generated code is based on a Java-based REST framework called Restlet [59].

All the process-entities in RESTfulBP are regarded as resources. The rectangles in the BPMN diagram represent units of work. Figure 9 also shows the annotations to the task resource, including task name, descriptive URI, allowed methods and media types of the representations. The role of every task needs to be identified to organise the role-based business rules. BPMN uses two types of swim-lane, pool and lane, to indicate the responsibility of different participants and to help partition and organise the tasks. Once a swim lane is annotated with a role, all tasks inside the lane own the same role.

State is an important process-entity. The State-based workflow patterns are supported within limits by nearly all existing flow-based business process languages and runtime engines because states are often internal shared data. Our annotation plugin explicitly supports the notion by mapping states as external addressable resources, which are annotated as a property of an outgoing edge of a task. It represents the state of a process instance or a milestone after a certain task is accomplished and before the subsequent tasks start. Business objects are annotated on the process model as artefacts.

The connectedness among tasks is essential for the coordination of distributed processes. Task connectedness indicates the dependency among the tasks. Querying the original process model can collect connectedness information. Once the BPMN diagram is active, every listed task has one xml file that contains the basic routing information of the task. The routing information is captured from the BPMN diagram. Further, more routing information files can be added to every task resource by adding an xml file with extra routing information. All of the different routing information can be represented in the generated code by maintaining a list of process fragments for every task resource. By using the tree structure of our modelling tool, the developer can easily generate several variants of one predefined process.

The process fragments generated from the BPMN-based process model are intended to be the obligation part of the process. The code generation engine also gathers the business rules in the annotation of tasks and generates the role resources as the repository of business rules. Using the rule-based support, the process variants and advanced workflow patterns are modelled by additional business rules, which can be further used to generate the corresponding process fragments automatically at runtime.

5.2. Runtime Engine

At runtime, the RESTfulBP process engine executes the business process that was produced. The process fragments to be executed are selected by the knowledge worker on the basis of personal experience and domain-specific knowledge. If a certain selected process fragment is completed, or the execution context matches any condition of the business rules, the RESTfulBP process engine would prepare the available process fragments for the knowledge worker.

RESTfulBP runtime engine extends the Restlet engine [59] with additional process-related functionality. Developing an extension to the Restlet engine can avoid handling the REST messages. The Restlet engine facilitates the processing of incoming and outgoing REST messages. Generally, the runtime engine of a traditional flow-based process definition language introduces message correlation and state maintenance mechanisms for the Web service composition. A message correlation mechanism is unnecessary in RESTfulBP because the process instance identifier is embedded into the task URI once the case is created. Thus, the execution of tasks in RESTfulBP is explicitly specified by the process instances. For the state
maintenance mechanism, the runtime engine of a traditional flow-based process specification language stores the internal state of the process in a centralised place. In RESTfulBP, however, every interesting piece of information the server can provide, such as process status, is exposed as a resource identified by a URI, which is called addressability [47]. Thus, in RESTfulBP, the state information is maintained distributed by the process-entity resources. The state information of a case during a task execution is exposed as a sub-resource of the task resource with the relative URI “/{taskname}/{caseId}”. Thus, RESTfulBP runtime engine can access the internal state through the HTTP GET method at any time without breaking the statelessness principle.

The process context information is stored as process fragments, which are exchanged during runtime among process participants. For the task with the <synchronisation> construct in its process context information, the RESTfulBP runtime engine checks the case-specific state of the corresponding tasks to join them when they are completed. Only the information in the <nextsteps> construct is returned to the client once the task is completed. Otherwise, all process context information is returned when clients request the process state during the task execution.

The information in the <condition> element (and <state> element in some situations) of the <rule> element inside the role resource is used to match the status of the business objects of a particular process instance. The process fragments have two parts. The static part is generated by the routing information from the corresponding task resources, while the dynamic part is generated at runtime according to the process context. The process fragments generated from BPMN contain only the obligation part, while those generated from the RNM-based business rules may contain both the obligation and permission parts. The generation of process fragments from the business rules is straightforward and mechanical. If a task is prohibited, it will not be included in the process fragment. The two parts of process fragments are combined and returned to the knowledge worker. With the rule-based support, the process fragments of both the basic process and the process variants can be generated automatically from either the BPMN diagram or the business rules.

6. Evaluation

The proposed approach is evaluated from two perspectives. First, we evaluated the expressiveness, usability and general usefulness of our approach by conducting a survey of our industry partners who have used this approach in a variety of systems and business process scenarios; the surveys reflect end-user evaluations. Second, we conducted a comparative adaptability analysis of RESTfulBP without and with the RNM-based rule support from the design perspective. The case used in this evaluation is from a real-world property-evaluation application we developed for a major property valuation company using the B2B process standard of the Australian lending industry. The system is used at this writing 24/7 by more than one hundred knowledge workers (property valuers, admins/task assigners, managers and corresponding bank staff). We examined other business processes used by our partners, but they are often complex and some are too commercially sensitive to be presented as another case. Additionally, we feel they do not add anything significantly new, technically, to the evaluation. Thus, we present a single case in our case study section. The systems that our partners used before our approach range from ad hoc to BPMN or similar workflow systems. The case study in the next section used Windows Workflow Foundations before switching to our system. Therefore, the case study and surveys reflect a comparison with a wide range of systems.

6.1. Expressiveness and Usability Survey of RNM-based Process Models/Fragments

6.1.1. Survey plan and execution

Survey purpose
This survey is an exploring survey. The main objectives of the survey are as follows. We use RNM as a shorthand for the RNM-based RESTful process approach.

1. To what types of business processes and application systems was RNM applied?
2. Evaluate the expressiveness and usability of RNM concepts and notations in specifying process models and fragments.

**Quantitative evaluation criteria**

Regarding our objectives, we relied on both literature and in-depth interviews to arrive at a set of criteria. The initial criteria largely come from [63, 64] around evaluation criteria of workflow meta-models. We interviewed four experts of workflow and business process management workflows with our early version of the survey and sought feedback. These four experts included one workflow researcher, one business process management consultant, one CIO from a government department and one CIO from a commercial company. Each interview lasted approximately 1.5 hours. We used the interviews to revise our criteria and survey design, and also reflected the opinions expressed in the final survey reporting. The overall criteria include RNM expressiveness (generality and clarity) and RNM usability (convenience, maintainability and ease of learning). We then proposed the following hypotheses.

H$_{1}$: The RNM can describe most of the business logic.

H$_{2}$: The RNM can describe the business logic clearly.

H$_{3}$: The RNM can be conveniently used for specifying business process models/fragments.

H$_{4}$: The business process models/fragments specified using RNM can be easily maintained.

H$_{5}$: The RNM can be easy to learn.

**Questionnaire design and Data collection**

Based on the above evaluation criteria and hypotheses, we designed a questionnaire. The main content of the questionnaire is shown below. Questions 1-3 collect personal information of respondents. Questions 4-8 measure the above five evaluation criteria. Question 9 mainly focuses on RNM users. Questions 10-12 are open questions that investigate the current application status, as well as advantages and disadvantages of RNM.

The above proposed quantitative evaluation criteria were measured on a five-point scale.

All participants were our partners who have used RNM for specifying process models/fragments or directly implementing systems in their work.

The sampling method is non-random sampling (mainly snowball sampling). The questionnaire was sent by email.

---------------------------------------------------------------------------------------------------------------------------------------

**Questionnaire about Role Network Model (RNM)**

1. Your position or role (at the time of using RNM) was ______.
2. Number of years of work experience ______.
3. Are you familiar with Role Network Model (RNM) ? ______
   A. Very familiar   B. Familiar   C. Generally understand   D. Understand a little   E. Do not understand
4. How well can the RNM describe the business logic that you want to express? ______
   A. All   B. Most   C. Half   D. Less than half   E. None
5. How clearly can the RNM describe the business logic that you want to express? _____
   A. Very clearly  B. Relatively clearly  C. Generally clear  D. Not very clear  E. Not clear at all

6. How convenient is the RNM to use for specifying business process models/fragments? _____
   A. Very convenient  B. Relatively convenient  C. Generally convenient  D. Not very convenient  E. Not convenient at all

7. Can the business process models/fragments specified using RNM be easily maintained? _____
   A. Very easily  B. Relatively easily  C. Generally easily  D. Not too easily  E. Not easily at all

8. Is the RNM easy to learn? _____
   A. Very easy  B. Relatively easy  C. Generally easy  D. Not very easy  E. Not easy at all

9. What would be the most suitable user groups for using RNM?
   A. Information management staff  B. Software engineer or business analysts  C. Business manager

10. Which business processes have you modelled by applying RNM and which RNM-backed systems have you implemented by applying RNM? How many users does the system have?

11. What are the advantages and disadvantages of a system based on RNM over other systems?

12. What are the advantages and disadvantages of the RNM in comparison with other traditional workflow models?

------------------------

6.1.2. Survey Result and Analysis

Sampling description

The respondents to the questionnaire include information management staff, software engineers, business analysts and researchers. We received 19 responses with 12 being valid (we excluded the 7 responses with D or E on question 3). Three of the respondents worked as information management staff for the government; four worked as software engineers or business consultants, and two as researchers. In addition, there were three respondents that have worked as both software engineer and researcher. The work experience of the respondents averaged 10.42 years. Table 2 provides a frequency summary of the answers provided by the respondents to questions 3-9.

<table>
<thead>
<tr>
<th>Option</th>
<th>Q 3</th>
<th>Q 4</th>
<th>Q 5</th>
<th>Q 6</th>
<th>Q 7</th>
<th>Q 8</th>
<th>Q 9</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>f %</td>
<td>f %</td>
<td>f %</td>
<td>f %</td>
<td>f %</td>
<td>f %</td>
<td>f %</td>
</tr>
<tr>
<td>A</td>
<td>4</td>
<td>33.3</td>
<td>3</td>
<td>25.0</td>
<td>5</td>
<td>41.7</td>
<td>8</td>
</tr>
<tr>
<td>B</td>
<td>5</td>
<td>41.7</td>
<td>8</td>
<td>66.7</td>
<td>6</td>
<td>50.0</td>
<td>4</td>
</tr>
<tr>
<td>C</td>
<td>3</td>
<td>25.0</td>
<td>1</td>
<td>8.3</td>
<td>1</td>
<td>8.3</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Quantitative analysis

Descriptive statistics of the answers to questions 4-8 are shown in Table 3.

| Variables      | Minimum | Maximum | Mean   | Std. Deviation |
|----------------|---------|---------|--------|----------------|----------------|
| Generality     | 3.00    | 5.00    | 4.1667 | 0.57735        |
| Clarity        | 3.00    | 5.00    | 4.3333 | 0.65134        |
| Convenience    | 4.00    | 5.00    | 4.6667 | 0.49237        |
| Maintainability| 3.00    | 5.00    | 4.5833 | 0.79296        |
| Understandability| 4.00 | 5.00    | 4.5833 | 0.51493        |
The One-Sample T Test method is used to obtain the conclusion whether the proposed five hypotheses (Hₐ-H₅) are accurate. However, the One-Sample T test general requirements the sample data generally matching the normal distribution. Thus, the One-Sample Kolmogorov-Smirnov Test is used to verify that the sample data of five evaluation criteria following the normal distribution, and the original hypothesis is that the test distribution is Normal. The results of the corresponding One-Sample Kolmogorov-Smirnov Test are shown in Table 4 and indicate whether the evaluation criteria distribution follow a normal distribution.

<table>
<thead>
<tr>
<th>Table 4. One-Sample Kolmogorov-Smirnov Test</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>N</strong></td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>Normal Parameters</td>
</tr>
<tr>
<td>Parameters</td>
</tr>
<tr>
<td>Most Extreme Differences</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Kolmogorov-Smirnov Z</td>
</tr>
<tr>
<td>Asymp. Sig. (2-tailed)</td>
</tr>
</tbody>
</table>

a. Test distribution is Normal.
b. Calculated from data.

As shown in Table 4, the Asymp. Sig. (2-tailed) of every evaluation criteria is greater than significance level 0.01. It shows that the corresponding original hypothesis cannot be rejected and the five evaluation criteria data all match the normal distribution with significance level 0.01. And then, with significance level 0.05, the One-Sample T Test results of evaluation criteria Mean equal 4 are shown in Table 5.

<table>
<thead>
<tr>
<th>Table 5. One-Sample T Test of evaluation criteria Mean equal 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variables</td>
</tr>
<tr>
<td>------------</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Generality</td>
</tr>
<tr>
<td>Clarity</td>
</tr>
<tr>
<td>Convenience</td>
</tr>
<tr>
<td>Maintainability</td>
</tr>
<tr>
<td>Understandability</td>
</tr>
</tbody>
</table>

As shown in Tables 3 and 5, Hₐ and H₆ are supported from the aspect of expressiveness. In detail, the mean of generality for Hₐ is 4.1667. Additionally, the result of a one-sample T test for generality, Sig. = 0.339 > 0.05, which illustrates there is no significant difference between mean of generality and 4. Therefore, Hₐ is supported, and RNM can describe most of the business logic. For H₆, the mean of clarity is 4.3333. In the one-sample T Test for clarity, Sig. = 0.104 > 0.05, showing there is no significant difference between mean of clarity and 4. Therefore, H₆ is supported; the RNM can describe the business logic relatively clearly.

For the aspect of usability, H₉, H₇ and H₈ are supported. In detail, for H₉, the mean of convenience is 4.6667 with Std. Deviation 0.49237; the minimum and maximum of convenience are 4 and 5. In the one-sample T test for convenience, Sig. = 0.001 < 0.05, which shows that there is a significant difference between the mean of convenience and 4, and the mean is greater than 4. Therefore, H₉ is supported, and the RNM can be relatively convenient when used for specifying business process models/fragments. For H₇, the mean of maintainability is 4.5833, Std. Deviation is 0.79296, and the minimum and maximum of
maintainability are 3 and 5. In the one-sample T Test for maintainability, Sig. = 0.027 < 0.05, which illustrates that there is significant difference between the mean of maintainability and 4, and the mean is greater than 4. Therefore, H₀ is supported, and the business process models/fragments specified using RNM can be relatively easily maintained. For H₀, the mean of understandability is 4.5833, Std. Deviation 0.51493, minimum and maximum 4 and 5. In the one-sample T Test for understandability, Sig. = 0.002 < 0.05, which means there is significant difference between the mean of understandability and 4, and the mean value is greater than 4. Therefore, H₀ is supported, and the RNM is relatively easy to learn.

Qualitative analysis

Based on the answers provided by the respondents to Question 9, 50% of respondents felt RNM is suitable for the third class of users, and the most suitable people are Information management staff. 25% of respondents believe RNM is suitable for Software engineers or business analysts, and the remaining 25% believe RNM is suitable for business managers.

Based on the answers provided by the respondents to Question 10, RNM has been applied in the following business process/fragment modelling situations: government office workflow management, Administrative permit process, University office workflow management, LIXI (Lending Industry XML Initiative). RNM has been applied to 22 application systems, including 9 government OA systems, 12 administrative permit systems, system of lending companies that following LIXI, etc. There are more than 1000 users in total. The above survey results show that RNM has good applicability and has been validated by real-world applications.

Based on the answers provided by the respondents to Question 11, the information systems based on RNM have 9 advantages and 4 disadvantages over other systems (such as Lotus Notes). And based on the answers provided by the respondents to Question 12, the RNM-based process models/fragments have 7 main advantages and 3 disadvantages over traditional workflow models. These answers are summarized as shown in Table 6.

<table>
<thead>
<tr>
<th>Question number</th>
<th>Advantage/Disadvantage</th>
<th>ID</th>
<th>Answers</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>Advantage</td>
<td>Q11A1</td>
<td>more suitable for complex systems</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q11A2</td>
<td>better description of the essence of business process coordination</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q11A3</td>
<td>better description of processes from a management perspective</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q11A4</td>
<td>can describe more business logic</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q11A5</td>
<td>can describe business processes more clearly</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q11A6</td>
<td>better support for dynamic business processes and flexible controls</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q11A7</td>
<td>makes business process reengineering easier</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q11A8</td>
<td>better alignment with real organisational structure and more focus on the allocation of roles and relationships, which make authority allocation and access control more convenient</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q11A9</td>
<td>at a higher degree of abstraction</td>
</tr>
<tr>
<td></td>
<td>Disadvantage</td>
<td>Q11D1</td>
<td>still lacks mature tools for visual design</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q11D2</td>
<td>needs better support for federated/distributed processes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q11D3</td>
<td>needs better support for communication and file processing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q11D4</td>
<td>the user community is still too small</td>
</tr>
<tr>
<td>12</td>
<td>Advantage</td>
<td>Q12A1</td>
<td>treats Role as first-class element in business process modelling and the business process is described through interaction between different roles, which better reflects the foundation of business coordination</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q12A2</td>
<td>can describe a complex business process better (generality), which is more suitable for changing scenarios</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q12A3</td>
<td>can express the business process more clearly</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q12A4</td>
<td>focuses on nodes (roles) of a network, while the traditional workflow model focuses on the edges (activities); thus, in many situation RNM is less complex than traditional workflow models</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q12A5</td>
<td>supports flexible management of dynamic business processes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Q12A6</td>
<td>easily maintained and adapted</td>
</tr>
</tbody>
</table>
In summary, RNM has been applied in a number of application systems and validated by real world scenarios. In addition, some above answers qualitatively support the related assumptions as shown in Table 7, and others show some additional advantages and weaknesses of RNM as shown in Table 8 and Table 9.

<table>
<thead>
<tr>
<th>Disadvantage</th>
<th>Q12A7</th>
<th>better reflects the relationships between staff and suitable for management of cross-organisation business processes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q12D1</td>
<td></td>
<td>less mature visualisation compared to traditional workflows</td>
</tr>
<tr>
<td>Q12D2</td>
<td></td>
<td>the theoretical aspect is not as well-understood as workflows</td>
</tr>
<tr>
<td>Q12D3</td>
<td></td>
<td>limited real world applications</td>
</tr>
</tbody>
</table>

Table 7. Qualitative support for the related hypotheses

<table>
<thead>
<tr>
<th>Hypothesis</th>
<th>Answer’s IDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>H_A</td>
<td>Q11A1, Q11A4, Q12A2</td>
</tr>
<tr>
<td>H_B</td>
<td>Q11A5, Q12A3</td>
</tr>
<tr>
<td>H_D</td>
<td>Q11A7, Q12A6</td>
</tr>
</tbody>
</table>

Table 8. Additional advantages of RNM

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Answer’s IDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naturalness</td>
<td>Q11A2, Q12A1, Q12A7</td>
</tr>
<tr>
<td>Flexibility</td>
<td>Q11A6, Q12A5</td>
</tr>
<tr>
<td>Less complexity</td>
<td>Q12A4</td>
</tr>
<tr>
<td>Integration with authority and access control</td>
<td>Q11A8</td>
</tr>
</tbody>
</table>

Table 9. Weaknesses of RNM

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Answer’s IDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visualization</td>
<td>Q11D1, Q12D1</td>
</tr>
<tr>
<td>Maturity</td>
<td>Q11D4, Q12D2</td>
</tr>
</tbody>
</table>

6.1.3. Survey validity discussion

Content validity: although this is an exploration survey, we determined five quantitative evaluation criteria through literature review and industry interview that ensure the content validity of the survey. However, quantitative evaluation criteria should be further improved in terms of comprehensiveness.

Measurement validity: this survey is not an verification study. Therefore, the number of samples is not large, which may affect the measurement results.

6.2. Scenario-Based Comparison of RESTfulBP without and with the RNM-Based Rule Support

6.2.1. Property Valuation Process

The property valuation process in the loan approval process is selected as case to illustrate our work. It comes from a part of the system we built and deployed for a major property valuation company, which has more than one hundred knowledge worker involved. We select the part for two reasons. Firstly, the property valuation process is a long running process, which requires adaptability to accommodate various scenarios and exceptions at both design time and runtime. The process also requires interactions among different roles, including lender, valuation firm and valuer. This process, therefore, forms a complicated business process with a mixture of human-intensive and automated tasks. Secondly, the property valuation process is a typical and representative business process running every day in the real world. It is also a part of the reference process definition for Australian lending industry [60, 61]. Some documents used in this case study are taken from the LIXI (Lending Industry XML Initiative) consortium, which is established as a non-profit, independent, member based industry organisation to develop e-
commerce standards to improve business-to-business interoperability and remove barriers to electronic data exchange within Australian lending industry.

Figure 10 shows the lifecycles of valuation instruction and fee negotiation. The property valuation process (shown on the left side) starts with an initial valuation request received by the valuation firm. The valuation request is processed through several processing stages until the valuation is completed and a response is sent back to the requestor. Every processing stage has a corresponding valuation status code to indicate the location of the instruction in the processing cycle. As well as the valuation instruction workflow, a simple workflow runs concurrently to manage requests for changes in the negotiated fee. This secondary workflow (shown on the right side) may be initiated from the state Accepted to state Inspected within the lifecycle of valuation instruction. The fee negotiation process runs concurrently to prevent the fee negotiation holding up the main valuation process.

![Diagram of valuation instruction and fee negotiation](image)

**Figure 10. Lifecycle of valuation instruction and fee negotiation**

Figure 11 shows the BPMN-based process model of the main valuation process mixed with the fee negotiation process. Three roles are identified in the valuation firm, namely, accountant, valuer and admin staff. The lender initiates the process by sending a valuation request to the valuation firm. Upon the receipt of a request, the corresponding valuation is Instructed. After the valuation is Accepted by the admin staff, the valuation is Assigned to a valuer. The actual valuation process is enclosed in the area with gray colour, the detail of which will be discussed later. The valuation needs the cooperation with the property owner. Ideally, the valuation state transfers through In-progress, Inspected during the valuation. However, exceptions may happen during the valuation, which can delay the valuation. The valuer’s work is completed once the initial valuation report is available for the admin staff to approve. The valuation is Completed when the final report prepared by the admin staff is ready to be sent back to the lender. At the same time, the accountant sends out the invoice. The payment from the lender can advance the valuation to Finalized state.
There are three kinds of inspections, namely, desk-top inspection, estimating the property value based on the existing data repository with a negligible cost; curb-side inspection, performing the external inspection and costs around $100 per property; and full inspection with both internal and external inspections and costs around $300. One or more types of inspections may be initiated for a certain process instance. For example, for some process instances, two of the inspection methods may be initiated at the same time, such as desk-top inspection and curb-side inspection, or desk-top inspection and full inspection. In the case that the property can be evaluated well enough on the basis of the existing data in the database and the transaction is urgent, the admin staff can approve the valuation with the result of the inspection that was completed first, ignoring the result of the other inspection. It does not matter which inspection is executed and completed first, although normally, the desk-top inspection takes less time than the other two types of inspections.

During the valuation process, the valuation fees may need to be renegotiated as a result of the complicated site conditions, such as irregular building shape and slopes. These complicated site conditions can be identified via every of the three kinds of inspections. Valuers are required to record the nature of the complications once they are identified. The fee change request can be sent by the admin staff at any time once a complication is recorded. However, the fee is not required to be renegotiated immediately in real world. For example, in a valuation for a certain property, the valuer finds a complicated site situation during the desktop valuation; he records it and goes for the curb-side valuation. During the curb-side valuation, another complicated site situation might be identified, which may cause the valuer to record another complication, and send fee negotiation for both complications afterwards immediately, or later in case more complication would be identified during a full inspection. In the case that no complication is identified during the curb-side valuation, the valuer may decide to send the fee

Figure 11. BPMN-based process model of fee negotiation
change request for the first complicated site situation. For some clients with good credit history, the fee negotiation can be performed even after the valuation reports are ready to be sent for authorisation. The fee negotiation scenario is a very flexible process in terms of when and how the fee should be renegotiated. The process is decided dynamically by the valuer in charge (knowledge worker) at runtime according to the situation and experiences. Such a flexible process is difficult to be modelled by the basic BPMN diagram without business rules. The process model would be very complicated if it covers all the possibilities.

6.2.2. RESTfulBP Approach

Model the Process

The process model specifies several aspects of the process, including: 1) all the tasks at different levels of abstraction, differentiating between manual and automated tasks; 2) three process participants, including lender, valuation firm and property owner; 3) three roles in the valuation firm, include admin staff, valuer and accountant; 4) valuation relative information transferred among the three process participants, for example, the required data from lender and the data included in the valuation report; 5) the control dependency among tasks; 6) the annotated state transition of valuation onto the process model.

Address Process-Entity by URI

RESTfulBP realises process-entities as a set of resources that are identified by declarative URIs. In the valuation process, the process is represented as a resource identified by the URI “www.restfulbp.com/lixi/valuationprocess/”. The URI of a case is created by putting the case ID at the end of the URI of process resource, for example, “www.restfulbp.com/lixi/valuationprocess/1234”. Tasks are modelled as resources. For example, the “accept valuation” task of admin staff is modelled as the Request Acceptance resource identified by the URI “www.restfulbp.com/lixi/valuationprocess/requestacceptance”. Every state of the property valuation process is also modelled as a resource. For example, the URI “www.restfulbp.com/lixi/valuationprocess/accepted” identifies a resource, which includes the entire list of valuations that are accepted by the valuation firm. This case study uses the six basic HTTP methods, including GET, POST, PUT, DELETE, OPTION and HEAD to manipulate the process-entity resources. There is no additional method.

Specify the Representation of Process-Entity

When a method on a process-entity resource is called, information about its status is returned. The information may contain meta-information or status information about an entity (task/process/case/state) and process coordination information such as possible next-steps and connections among relevant tasks. The representation of process resource provides the specification of the valuation process, includes the method and the schema to execute the process. Similarly, the representation of task resource provides the specification of the corresponding task. The other two process-entity resources provide different representations according to different roles. A case resource has two kinds of representations. One represents the general information of the current status of a case, which is often returned by the GET method to a process participant. The other kind of output includes all possible next-steps for the participant who is permitted to advance the case. This information can only be returned to the relevant counterparties with appropriate authorization so that they can advance the case by requesting the PUT method on the URI provided in each possible next-step. Similarly, the state resource also has two kinds of representations. One represents all the case resources in the corresponding state without any executable hyperlink-based process fragments. The other one includes all the executable process fragments for a human with the role that is permitted to precede the cases.
Using the HTTP content negotiation, authentication and authorization mechanisms, the server can authorize the participant based on the specified roles, and return the corresponding information. An appropriate representation of a process-entity resource for the requestor is selected in accordance with requestor's role, which is indicated by an extended media type. For example, the resources of all the three tasks in the admin staff lane have the presentations of “Application/HTML/admin” type. The “delay valuation” task is an exception. Although the task is in the lane of valuer, practically, a client with any of the three roles in the valuation firm can delay the valuation once exception occurs during runtime.

*Generate Process Fragment*

The task resources are connected by using the XLink-based communication format. Part of the routing information is extracted from the BPMN model; the other part is defined manually. For example, to implement a state-based control-flow pattern, a state resource called Complicated Condition Identified” was implemented with the URI “/lixi/valuationprocess/complicatedcondition identified” to store all the valuations whose corresponding property has complicated condition (after the task “Record nature of complication”). The routing of the state resource is as below:

```xml
<nextsteps para = "1">
  <task xlink:type="simple" xlink:title="request fee change"
       xlink:href="/lixi/valuationprocess/feechangerequest/{caseId}"
       xlink:role="www.restfulbp.com/lixi/valuationprocess/valuer" method = "PUT"/>
</nextsteps>
```

Thus, the URI of Fee Change Request task resource is added into the routing information of every case at the state.

*Implementation*

Figure 12 gives three screenshot of a real world property valuation system that implemented our research results. The system contains a Web-based system and a PDA-based system. The web page on the top shows the administration UI for admin staff (Manager role), which lists all the property valuations progressing at different steps. The two screenshots on the bottom show how valuers (Valuer role) carry out the knowledge-intensive tasks. The left part shows a list of valuation tasks assigned to the valuer. The right part shows the screen that the valuer uses to input the details of the property being valuated. The two systems are connected through a RESTfulBP system.
6.2.3. RNM-based RESTfulBP Approach

**Model the Process**

In RNM-based process modelling, the BPMN is still used to model the structured part of the process. The RNM-based business rules are used as a complement to model the part of the process with more complexity, such as the part of valuation process enclosed in the grey area in Figure 11. Figure 13 shows the BPMN-based process model for the RNM-based RESTfulBP approach. Firstly, the tasks of the three inspections, the record nature of complication, and the delay valuation are wrapped in a rule-based sub-process with the ad hoc marker (tilde symbol), which indicates that the valuation sub-process is composed of a group of tasks that have no pre-definable sequence relationships [55]. The BPMN specification prescribes that the sequence and number of executions of the tasks inside an ad hoc sub-process is completely determined by the performer of the tasks and cannot be defined beforehand. Obviously, the semantic of the ad hoc cannot fully match the real scenario of valuation process. The valuation sub-process is modelled as a rule-based sub-process, which uses a set of business rules to add extra restrictions on the valuation sub-process. This largely simplifies the original model in Figure 11, and makes the process model clearer to read and easier to change. Secondly, the basic BPMN model, as presented in Figure 11, cannot model a scenario that requires the concept of state. Both RESTfulBP and RNM-based rules use the state concept explicitly as a first-class element in process modelling, and thus directly support the state-based workflow patterns. However, in RESTfulBP, the process fragments for state-based workflow patterns must be manually added to the representation of state resources at design time. In RNM-based RESTfulBP, the modelling of the state-based workflow patterns is on the basis of business rules, and consequently, the corresponding process fragments can be automatically generated at runtime.

In our case study, the valuation report is done by the valuer, including data of both the valuation and fee negotiation. Thus, two fine-grained business objects are identified, namely, valuation report (vr) and fee negotiation document (fnd). The valuation report records the detail of the property, the valuation progress and the estimated value. The fee negotiation document records the fee nominated by the requestor, the fee agreement and so on. The state transitions of the business objects are defined in Figure 8. Additionally, three roles are identified in the valuation firm, namely, administrative (admin) staff, valuer and accountant. In RNM approaches, role is a first-class concept rather than an attribute of other process-entities. Further information of role needs to be specified during the process modelling, such as operations and authorisation of the role, and business rules that define the relationship between the roles and business objects. Tables 10 and 11 give the RNM model for the valuation process. We use tables to show the business rules for illustration only; in our real system, the business rules are specified in XML format. Every row of Table 10 and Table 11 is an improved NAM-based business rule. This paper limits the discussion to the operations in the valuation firm. In the main process, the valuation report is controlled by the valuer. The business processes can be viewed as the value evolutions of business objects through state transitions. State variable $s_1$ and $s_2$ are used together to represent the state of the valuation report. The value sets of $s_1$ and $s_2$ are defined in the second and third rows of Table 10. Similarly, in the fee negotiation process, the fee negotiation document is allowed to be controlled by the admin staff. The state of the fee negotiation document is represented by the state variable $S_3$. The value set of $S_3$ is shown in the second row of Table 11.
Figure 13. BPMN model of the valuation process with RNM-based rules

The behaviour of a human with an authorised role is modelled as the transitions among the states of the business objects in terms of operations. Every activity in the BPMN model in Figure 10 is an operation that is allowed to operate on the corresponding business objects. Thus, the valuation report has six operations while the fee negotiation document has three operations. To describe the procedure of the valuation report more accurately, and also to show the relationships and interactions of the valuation process and fee negotiation process, more fine-grained states are introduced, such as Complication identified and Complication recorded.
Table 10. RNM model of the main process

<table>
<thead>
<tr>
<th>Business Object Name: valuation report</th>
<th>Business Object ID: vr</th>
<th>Role: Valuer</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_1 \in {\text{Assigned, Inspected, Delayed, Awaiting authorisation}}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$S_2 \in {\text{Complication identified, Complication recorded, null}}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Initial state: $S_1 = \text{Assigned} &amp; S_2 = \text{null}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>End state: $S_1 = \text{Awaiting authorisation} &amp; S_2 = \text{Complication recorded} | \text{null}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operation</th>
<th>Condition</th>
<th>Deontic Logic</th>
<th>Post-Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desktop inspection</td>
<td>$S_1 = \text{Assigned}$</td>
<td>Permitted</td>
<td>$S_1 = \text{Inspected} | \text{Delayed}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$S_2 = \text{Complication identified} | \text{null}$</td>
</tr>
<tr>
<td>Curbside inspection</td>
<td>$S_1 = \text{Assigned}$</td>
<td>Permitted</td>
<td>$S_1 = \text{Inspected} | \text{Delayed}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$S_2 = \text{Complication identified} | \text{null}$</td>
</tr>
<tr>
<td></td>
<td>$S_1 = \text{Inspected}$</td>
<td>Permitted</td>
<td>$S_1 = \text{Inspected} | \text{Delayed}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$S_2 = \text{Complication identified} | \text{null}$</td>
</tr>
<tr>
<td>Full inspection</td>
<td>$S_1 = \text{Assigned}$</td>
<td>Permitted</td>
<td>$S_1 = \text{Inspected} | \text{Delayed}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$S_2 = \text{Complication identified} | \text{null}$</td>
</tr>
<tr>
<td></td>
<td>$S_1 = \text{Inspected}$</td>
<td>Permitted</td>
<td>$S_1 = \text{Inspected} | \text{Delayed}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$S_2 = \text{Complication identified} | \text{null}$</td>
</tr>
<tr>
<td>Record complication</td>
<td>$S_2 = \text{Complication identified}$</td>
<td>Obliged</td>
<td>$S_1 = \text{Inspected}$ $S_2 = \text{Complication recorded}$</td>
</tr>
<tr>
<td>Solve delay</td>
<td>$S_1 = \text{Delayed}$</td>
<td>Obliged</td>
<td>$S_1 = \text{Assigned} | \text{Inspected} | \text{Awaiting authorisation}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$S_2 = \text{any value}$</td>
</tr>
<tr>
<td>Request authorisation</td>
<td>$S_1 = \text{Inspected}$</td>
<td>Permitted</td>
<td>$S_1 = \text{Awaiting authorisation}$</td>
</tr>
<tr>
<td></td>
<td>$S_1 = \text{Inspected}$</td>
<td>Permitted</td>
<td>$S_2 = \text{Complication recorded} | \text{null}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$S_1 = \text{Awaiting authorisation}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$S_2 = \text{Complication recorded} | \text{null}$</td>
</tr>
<tr>
<td></td>
<td>$S_1 = \text{Awaiting}$</td>
<td>Permitted</td>
<td>$S_1 = \text{Awaiting authorisation}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$S_2 = \text{Complication recorded} | \text{null}$</td>
</tr>
</tbody>
</table>

Table 11. RNM model of the fee negotiation process

<table>
<thead>
<tr>
<th>Business Object: fee negotiation document</th>
<th>Business Object ID: fnf</th>
<th>Role: Admin staff</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_2 \in {\text{null, Requested, Accepted, Rejected, Cancelled}}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Initial state: $S_3 = \text{null}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>End state: $S_3 = \text{Accepted}</td>
<td></td>
<td>\text{Rejected}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operation</th>
<th>Condition</th>
<th>State</th>
<th>Deontic Logic</th>
<th>Post-Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request fee change</td>
<td>$v_rS_1 = \text{Assigned}$</td>
<td>$v_rS_2 = \text{Complication recorded}$</td>
<td>Permitted</td>
<td>$S_3 = \text{Requested}$</td>
</tr>
<tr>
<td></td>
<td>$v_rS_1 = \text{Inspected}$</td>
<td>$v_rS_2 = \text{Complication recorded}$</td>
<td>Permitted</td>
<td>$S_3 = \text{Requested}$</td>
</tr>
<tr>
<td></td>
<td>$v_rS_1 = \text{Awaiting}$</td>
<td>$v_rS_2 = \text{Complication recorded}$</td>
<td>Client is not golden</td>
<td>Obliged</td>
</tr>
<tr>
<td></td>
<td>$v_rS_1 = \text{Awaiting}$</td>
<td>$v_rS_2 = \text{Complication recorded}$</td>
<td>Client is golden</td>
<td>Prohibited</td>
</tr>
<tr>
<td>Fee negotiation</td>
<td>$S_3 = \text{Requested}$</td>
<td></td>
<td>Permitted</td>
<td>$S_3 = \text{Accepted}</td>
</tr>
</tbody>
</table>

**Addressing the Process-entity by URI**

As above, every process-entity is modelled as a resource identified by a declarative URI. All the process-entity resources are controlled through the six basic HTTP methods. The URIs of the four basic types of process-entities are still the same as those in the RESTfulBP approach. The business objects are also
modelled as resources and identified by declarative URIs; for example, the URI of the valuation report is "www.restfulbp.com/lixi/valuationprocess/1234/bo/valuationreport".

**Specify the Representation for Process-entity**

The representations of the four types of process-entities are the same as in RESTfulBP. The representations of RNM-related resources follow the formats discussed in section 4.2. For example, the resource of the "fee negotiation document" task may have the representation below for admin staff after a complicated condition is identified and recorded. The operation of "fee negotiation" does not have an operation link because the operation is not active.

```xml
<bo>
  <id>fnf</id>
  <description>a document that assist the fee negotiation</description>
  <state>
    <variable>
      <variablename>S1</variablename>
      <variablevalue>Assigned</variablevalue>
    </variable>
    <variable>
      <variablename>S2</variablename>
      <variablevalue>Complication recorded</variablevalue>
    </variable>
  </state>
  <operationset>
    <operation>
      <operationname>Request fee change</operationname>
      <operationlink>www.restfulbp.com/lixi/valuationprocess/feechangerequest</operationlink>
    </operation>
    <operation>
      <operationname>Fee negotiation</operationname>
    </operation>
  </operationset>
</bo>
```

Another code snippet below shows the representation of the “admin staff” resource, storing the business rules in Table 11.

```xml
<role>
  <id>Admin</id>
  <type></type>
  <org>valuation firm</org>
  <sup></sup>
  <BOL>
    <id>fnf</id>
    <OPERL>
      <operation>Request fee change</operation>
      <operation>Fee negotiation</operation>
    </OPERL>
  </BOL>
  <OPERC>
    <rule>
      <boid>fnf</boid>
      <condition>vr:S1=Assigned; vr:S2=Complication recorded</condition>
      <state></state>
      <deontic>Permitted</deontic>
    </rule>
  </OPERC>
</role>
```
<operation> Request fee change </operation>
<post-condition> S3=Requested </post-condition>
</rule>

There were three rule-based task resources and one rule-based sub-process resource in our case study. The code snippet below shows the part of business rules in the representation of the Full Inspection task resource. It was dynamically generated from the rules repository.

<operation>
  <boid> fnd </boid>
  <condition> vr:S1=Awaiting authorization vr:S2=Complication recorded </condition>
  <state> Client is golden </state>
  <deontic> Prohibited </deontic>
  <operation> Request fee change </operation>
  <post-condition> S3=Requested </post-condition>
</rule>

<operation> Fee negotiation </operation>
<post-condition> S3=Accepted||Rejected||Cancelled </post-condition>

Generate Process Fragment
In RNM-based RESTfulBP, part of the routing information is generated from the BPMN-based model as before, while the rest is defined by RNM-based rules. Thus, the corresponding process fragments are generated based on both the diagram-based information and rule-based information. The routing information of normal tasks is generated from the BPMN-based basic process at design time as we discussed in our previous work [4]. All the process fragments generated from the BPMN model are included in the <obligation> element. For example, the process fragment of the Approve task resource is shown in the code snippet below. Value 2 of “para” means the two tasks could executed concurrently.

```xml
<nextsteps>
<obligation>
  <para>2</para>
  <task xlink:type="simple" xlink:title="Send invoice" xlink:href="/sendinginvoice/1234 " xlink:role="valuationprocess/accountant" method="PUT"/>
  <task xlink:type="simple" xlink:title="Prepare final report" xlink:href="/reportpreparation/1234 " xlink:role="valuationprocess/admin" method="PUT"/>
</obligation>
<nextsteps>

The process fragments of rule-based tasks are generated according to the corresponding business rules defined in the role resources. These parts of process fragments are dynamically generated at runtime. During the process execution, when a task is completed, the process engine generates the dynamic process fragments on the basis of the business rules and the task ID and execution status of the process instance. For example, in the valuation process, if a complicated condition is identified during the desktop-inspection and is recorded afterwards, the state of the process valuation report would be “S1=Inspected; S2=Complication recorded”. According to Table 10, there are three tasks in total that have “S1=Inspected” as pre-state. Thus, after the desktop-inspection, the process fragment returned for the valuer role is as shown below.

```xml
<nextsteps>
<permission>
  <task xlink:type="simple" xlink:title="Curbside inspect" xlink:href="/curbsidesinspection/1234 " xlink:role="valuationprocess/valuer" method="PUT"/>
  <task xlink:type="simple" xlink:title="Full inspect" xlink:href="/fullinspection/1234 " xlink:role="valuationprocess/valuer" method="PUT"/>
  <task xlink:type="simple" xlink:title="Request authorization" xlink:href="/authorizationrequesting/1234 " xlink:role="valuationprocess/valuer" method="PUT"/>
</permission>
<nextsteps>

According to Table 10, after the desktop inspection, if the client of this process instance is not a golden client, the "request fee change" task would be included in the process fragment returned to the admin role as permission; otherwise, the fee negotiation can be skipped. Assume that in a certain instance the client is not a golden client, and the valuer chooses to do a curbside inspection after the desk-top inspection. The execution of a curbside inspection cannot change the state of the valuation report. The "curbside inspect" task is removed from the process fragment after it is executed. However, the execution of curbside inspection does not affect the process fragment for the admin staff. The valuer decides to request authorisation after completing the curbside inspection, which changes the state of valuation report into “S1=Awaiting authorisation; S2=Complication recorded” and empties the process fragment for the valuer role. At this moment, the process fragment for the admin role is also changed, as shown below. The "Request fee change" task is no longer permission, but obligation for the admin staff. The admin staff can also do some management tasks throughout the valuation process, such as editing the rules for the valuer assignment and valuer characteristics. Thus the “edit rules for the valuer assignment” task and the “edit valuer characteristics” task are in the permission part of the process fragment for the admin role throughout the valuation process.
6.2.4. Support of workflow patterns

The integration of business rules with RESTfulBP can improve the support of workflow patterns [51]. Workflow patterns provide a conceptual basis and evaluation framework for the business process technologies, such as process modelling methods and service orchestration languages. BPMN lacks support for the notion of state; thus, most state-based workflow patterns cannot be easily captured. In the original RESTfulBP, a new annotation was introduced into the process model to explicitly model state. However, the annotation increases the complexity of the process model. By using the state as the first-class concept in the business rules, our new approach does not model the state explicitly on the diagram, while still supporting all the state-based patterns. The fee negotiation sub-process in our example represents the milestone pattern [51]. Once a complicated condition is identified and recorded, the fee negotiation sub-process can be enabled. The sub-process cannot be enabled until the process instance has progressed beyond the state. For example, if the client that requests the process is a golden client, the fee negotiation sub-process can be skipped and cannot be enabled at a future time unless another complicated condition is recorded. There are several workarounds for this pattern using traditional BPMN, such as using a pair of start and end events of link type to connect the milestone and the nominated task [62], or using a separate branch to take care of the communication and the inquiry of the state of the process [52]. However, none of the existing solutions can exactly reflect the semantics of the milestone pattern. Using our approach, the milestone pattern is simply modelled by a single business rule, which explicitly indicates the nominated tasks for a milestone in terms of condition.

RESTfulBP assumes the process is often manually driven by a knowledge worker; thus, RESTfulBP does not have any mechanism to make an automatic decision at process splits. It simply provides all the possibilities to the knowledge worker, and lets the knowledge worker make the decision and drive the process instance. This is not enough for many revised workflow patterns [26], including many advanced patterns with complicated conditions. Using business rules to describe the routing information can easily support the advanced patterns because most of the complicated conditions can be put into the business rules and handled by the business rule engine. Although the scenario discussed in the case study only shows a limited number of workflow patterns, the integration of RNM-based business rules with RESTfulBP can improve the support of workflow patterns. Due to the length of the paper, only one revised workflow pattern [26] is discussed as an example.

Cancelling partial join is one of the Advanced Branching and Synchronisation Patterns [26]. It is the point when m branches join into a single subsequent branch. The thread of control is passed when n (n < m) of the incoming branches have been enabled. Triggering the join cancels the execution of all of the other incoming branches. For example, in our property valuation process, more than one valuer (normally two) might be assigned for the same valuation instruction for an urgent case. The admin staff uses the first valuation result completed, and cancels the valuation task for the other valuer. A model of this pattern is shown in Figure 14. Table 12 shows the RNM-based business rules included in the Task_rule that count the number of completed tasks and cancel the remaining tasks according to the condition.
domains and synchronisation of business rules are still not resolved. We plan to evaluate our approach in other capability using business rules is weak. For cross-adaptive and the development effort is reduced through the automatic generation of process fragments at runtime using RNM rules. However, the current approach has several limitations. The exception handling capability using business rules is weak. For cross-organisational processes, the storage and the synchronisation of business rules are still not resolved. We plan to evaluate our approach in other domains and to assess the extent to which core process models can be adapted at runtime.

6.2.5. Complexity

BPMN has limited expressiveness to describe a complicated split such as that to initiate different (types of) inspections. It is the valuer (knowledge worker) who decides on one or multiple inspections at runtime to be initiated for a certain process instance. This scenario can be modelled by the complex gateway construct of BPMN. However, the scenario in the real world is more complicated in that the different inspections can be executed sequentially or in parallel. For example, in the valuation of a certain property that initiates a desk-top inspection and a curbside inspection, the full inspection could be executed afterwards if the admin staff is not satisfied with the results of both inspections. This complicated scenario includes the semantic of several workflow patterns of branching and synchronising. The BPMN model in Figure 9 uses several complex gateways, such as event-based gateways and conditional flows, to model such a complicated scenario. However, how the control actually flows in such an intricate model is unclear. Additionally, this model still cannot express the real world scenarios. In the implementation of the original RESTfulBP approach, the process fragment of every task within the valuation sub-process includes all the tasks within this sub-process. The knowledge worker is the one who decides the execution sequence of the tasks. This approach largely relies on the knowledge worker, and has no mechanism to constrain the behaviour of knowledge worker. Expressing the control-flow in the business rules can decrease the process complexity, while introducing some constraints into the routing information for knowledge workers.

7. Conclusions and Future Work

Our previous work, RESTfulBP, aligns the REST architectural style with Web-based business processes. This style uses process fragments to enable process coordination and choices at the endpoint, especially for human-intensive Web-based business processes, to improve system adaptability. However, this style was ineffective in expressing choices and auto-generation of process fragments in declarative ways. RNM is a role-based business rule approach that provides a more compressive declarative style for modelling business processes. This paper introduces RNM-based rule support for RESTfulBP. This support increases the expressiveness of RESTfulBP and adds extra deontic logic constraints to the process fragments to restrict the behaviour of knowledge workers. The evaluation indicates that the resulting system is more adaptive and the development effort is reduced through the automatic generation of process fragments at runtime using RNM rules. However, the current approach has several limitations. The exception handling capability using business rules is weak. For cross-organisational processes, the storage and the synchronisation of business rules are still not resolved. We plan to evaluate our approach in other domains and to assess the extent to which core process models can be adapted at runtime.
Acknowledgement

This research is partially supported by NICTA through the ICT Centre of Excellence program funded by the Australian Research Council and the Department of Communications funded by the Australian Government, and by National Natural Science Foundation of China (Grant No.71001013), and by the Specialized Research Fund for the Doctoral Program of Higher Education of China (Grant No.200801411060), and by the Fundamental Research Funds for the Central Universities of China (Grant No. DUT09RC(3)060) and by “the Fundamental Research Funds for the Central Universities” (No. 14CX02140A and No.14CX02137A) and “the Scientific Research Foundation of China University of Petroleum” (No. Y1307021).

References

11. X. Xu, I. Webber, L. Zhu, J. Liu, P. Rimba and Q. Lu, "BPMashup: Dynamic Execution of RESTful Processes", The 10th International Conference on Service Oriented Computing (ICSO2012), Shanghai, China
27. G. Steinkne, C. Nickolette, Business rules as the basis of an organization's information systems, Industrial Management & Data Systems, 103(1) 2003, pp. 52-63.
40. F. Rosenberg, S. Dustdar, Business rules integration in BPEL - a service-oriented approach. The 7th IEEE International Conference on E-Commerce Technology (CEC 2005), Technische Universität München, Germany, pp. 476 - 479
47. L. Richardson, S. Ruby, RESTful Web Services, 2007, O'Reilly.


Vitae

Xiwei XU received her BEng degree in Software Engineering from Nankai University (NKU), in Tianjin, China, in 2007. She received her Ph.D. degree in the School of Computer Science and Engineering at University of New South Wales (UNSW), in Sydney, Australia, in 2011. She is also with National ICT Australia (NICTA). Her research interests is in software architecture, business process and cloud computing.

Xin Ye is an associate professor at Dalian University of technology (DUT). He received his Ph.D. degree at Dalian University of Technology (DUT), in Dalian, China, in 2006. His research interests include business process modelling and analysis, management information system and software engineering.

Liming ZHU is a principal researcher at NICTA (National ICT Australia). He holds conjoint academic positions and teaches software architecture courses at both University of New South Wales (UNSW) and The University of Sydney. His research interests include software architecture, service engineering and system development methodologies. He has published more than sixty scientific papers. He contributes to software architecture related standards and technologies at both ISO and several industry consortiums. He formerly worked in several technology lead positions in software industry before completing a PhD degree in Software Engineering at UNSW.

Qinghua LU is a lecturer at the College of Computer and Communication Engineering, China University of Petroleum. She received her PhD from University of New South Wales in 2013. Her research interests include software architecture, dependability of distributed systems, and business-driven IT management. She is a member of the IEEE Computer Society.