Service Oriented Computing and Applications (2018) 12:25–39
https://doi.org/10.1007/s11761-018-0233-5

SPECIAL ISSUE PAPER

Resilience analysis of service-oriented collaboration process management systems

Paul de Vrieze[1] · **Lai Xu**[1]

Received: 7 February 2017 / Revised: 28 February 2018 / Accepted: 6 March 2018 / Published online: 27 March 2018

Abstract
Collaborative business process management allows for the automated coordination of processes involving human and computer actors. In modern economies, it is increasingly needed for this coordination to be not only within organizations but also to cross organizational boundaries. The dependence on the performance of other organizations should, however, be limited, and the control over the own processes is required from a competitiveness perspective. The main objective of this work is to propose an evaluation model for measuring a resilience of a service-oriented architecture (SOA) collaborative process management system. In this paper, we have proposed resilience analysis perspectives of SOA collaborative process systems, i.e., overall system perspective, individual process model perspective, individual process instance perspective, service perspective, and resource perspective. A collaborative incident and maintenance notification process system is reviewed for illustrating our resilience analysis. This research contributes to extend SOA collaborative business process management systems with resilience support, not only looking at quantification and identification of resilience factors, but also considering ways of improving the resilience of SOA collaborative process systems through measures at design and runtime.

Keywords Collaborative business processes management · Resilience · Resilience of SOA collaborative process systems · Service-oriented systems · Cloud-based SOA collaborative business process management

1 Introduction

Collaborative business processes are increasingly driven by business flexibility and agility. The increasing importance of value chains and production networks, of interconnected organizations, collaboration dynamics, outsourcing, and the increasing potential of new ICT technologies supported innovations have driven research into such collaborative networks [33]. Collaborative networks [6,12] and collaborative business process management [27,33] have been fostered by globalization over decades.

Advances in ICT, Internet, and cloud computing have led to the explosion of collaborative networks both at organizational and individual contributor levels. A pervasive interconnection of, and collaboration among, physical and virtual objects provides opportunities for the realization of manufacturing 2.0 (industry 4.0) and sustainable development, but only as long as this does not undermine flexibility and resilience.

Collaborative networks as a relatively new scientific discipline [6] have been applied to application areas such as Factories of the Future [13] as well as manufacturing and logistics networks [3]. Organizations, enterprises, and communities are interconnected by networks in the new application areas. To support such collaboration in a hyper-connected world, existing technologies need to be improved and adapted in terms of larger-scale integration and more intelligent devices, sensors and cyber-physical systems involvement. The current business environment is challenging. As a result, systems need to support enterprise agility and be resilient in turbulent business environments [5].

The concept of resilience has gained importance for engineered systems as a way to address their complexity [22].

✉ Paul de Vrieze
pdvrieze@bournemouth.ac.uk

Lai Xu
lxu@bournemouth.ac.uk

[1] Faculty of Science and Technology, Bournemouth University, Talbot Campus, Fern Barrow, Poole, Dorset BH12 5BB, UK

The systems are not only designed to be more reliable, but are also required to be more resilient to withstand unanticipated failures without catastrophic losses [14,28]. Different disciplines have different definitions of resilience. In materials science, resilience represents the ability of a material to recover its original shape following a deformation [7]. In the corporate world, resilience refers to the ability of a company to bounce back from, or even resist, a large disruption—this includes, for instance the speed with which it returns to normal performance level (production, services, fill rate, etc.) [8]. Within cloud computing, the characteristic of resiliency can refer to redundant IT resources within the same cloud (but in different physical locations) or across multiple clouds [29].

The use of different definitions of resilience in different fields leads to different method for measuring system resilience. Independent of this, in a traditional approach to process management, the process management system forms a single point of failure.

As, in general, collaborative process systems are critical for businesses, the lack of a systematic way to analyze the resilience of SOA collaborative process systems motivates our research. The research questions thus are to discuss an appropriate definition of resilience of SOA collaborative process management systems, as well as how to analyze, measure and improve the resilience of these systems.

In the context of SOA collaborative process management systems, resilience is defined as the degree as to which the system can continue to meet its stakeholders' expectations (or the goals of the system) in the presence of errors. Based upon an analysis of different perspectives of SOA collaborative process management systems, we have analyzed how the system resilience can be improved in SOA collaborative process management system from different perspectives.

In this research, we are especially considering how to manage inter-enterprise collaboration, i.e., collaborative business processes that are executed among enterprises and which cross the organizational boundaries. Within these collaborations, we not only look at the quantification and identification of resilience factors, but also consider ways of improving the resilience of SOA collaborative process systems through measures at design and runtime. Within this context, we consider that (web) services and human interactions are coordinated through an automated process management system to form an SOA collaborative process management system.

Our interpretation of an SOA collaborative process system is one where starting from a single-organization perspective a process management system is used to coordinate activities that are normally executed as services implemented by multiple constituent systems (whether a service interface to a classical system or a microservice). This approach is then expanded through.

In this paper, we focus on SOA collaborative business process systems as well as analysis of resilience related the different perspectives of SOA collaborative business process systems. This work is structured such that after introducing service-oriented collaborative process management in Sect. 2, Sect. 3 discusses the various applicable failure modes and analysis dimensions. Section 4 applies this to present an analysis of the different perspectives of SOACBPMS resilience. The resulting framework is then, in Sect. 5, applied to an incident notification case. Section 6 presents additional related work, leading to a conclusion in Sect. 7.

2 SOA collaborative process management

Collaborative business processes exist not only in a single company between different departments or divisions, but also among different organizations. Niehaves et al. [33] highlight that most collaborative business processes can be found from global value chains [16,39], production networks [40], interconnected firms [25], collaboration dynamics [2], outsourcing [20], etc. All mentioned examples involve business processes that have been created cross-organizationally. Research fields such as business process management, workflow systems, and computerized information systems consider design, deployment and implementation-related information systems to support above-mentioned applications, but do not focus on application in a collaborative context.

In a collaborative context, contract- or agreement-based collaboration approaches have been broadly used to support virtual enterprises [17,49], supply chains [31], inter-organizations [34], and e-commerce [23,24,50]. Collaborative processes can be also supported by service orchestration and choreography [26,47] among different organizations.

The SOA collaborative structure could be a peer-to-peer structure, a centralized structure, or a federated structure. In a peer-to-peer structure, the collaborative business process is implicitly defined by the process instances that drive sequencing within the partner's processes and by the interaction between the public operations of processes on all partners' sides [26]. The collaborative business process models specify the necessary synchronization among services which are involved in collaborations among partners.

In contrast, a centralized collaborative structure normally specifies the interactions among involved partners in an overall collaborative process model. The centralized structure is typically supported by pre-agreed interactions, i.e., a collaborative contract, a collaborative agreement. The overall process is decomposed into activities that are performed by individual partners, either implemented by an automated system (web service) or as a task for a human to perform (intermediated through some task management system). The

performance of actions can also be implemented as a subprocess, possibly one that remains within the confines of a single partner. These subprocesses can be explicitly incorporated in the overall process system or completely hidden as implementation detail invisible from the other partners (or their systems). Note that this does not consider tangential processes triggered by action execution, for example a commitment to deliver a good triggering an order process due to stock levels dropping below the threshold. When subprocesses are present, but not managed/coordinated by the central process management system this may be a case of a federated process.

Considering federated processes, they are distinguished from centralized process management in that process-aware collaboration occurs, but is controlled by multiple autonomous management entities. While federated process execution is normal when processes are managed and executed by humans (each person manages his own actions within the agreed boundaries), such coordination is not the norm for automated process management.

Federated process management can perhaps be best defined through counter cases. First, a single process management system that does not touch other process management systems (in any way at all) is not federated. Perhaps more interesting is the contrast with the scenario where two organizations (A and B) have their own process management system, and the execution of a process by organization A happens to involve a process (or multiple) in organization B, all without awareness. From the perspective of organization A, the process merely consumes a service provided by B. The processes (in A and B) may very well be mostly unrelated or tangential (the service consumption may trigger a CRM update in B's CRM). Federated process management in contrast implies that the process management systems have some sort of coordination and awareness. The form this takes can differ according to the process and business context.

As such, federated process management requires some degree of meta-level coordination between the process management systems that perform the service choreographies. Considering multiple copies of the same process (description) to be the same process (model), multiple instances of a single process model can, from a technical perspective, be instantiated on/coordinated by different management systems. For overall management it is, however, worthwhile that meta information is coordinated to allow for overall process-related statistics and information to be collected. Each process instance should have a single, eventually consistent, state.

When parts of a process are coordinated by different process management systems in a federation, there are different ways this can be organized. In this paper, we only consider those approaches where at any point of time there is a single "master" or "owner" of the process instance. The coordination of a subset of the activities can, however, be delegated to a different management engine. Ownership may also be transferred (voluntarily or in response to failure).

Processes that involve multiple parties may be collaborative. While it is possible for this to be limited to two parties, generally one would expect more than two parties. Various degrees of collaborativeness can be distinguished. At the lowest level of collaborativeness, collaborative processes have little differences to distinguish them from simple provision of (web) services. At the highest end, there are complex interactions between the involved parties.

In addition to degrees of collaborativeness, there is the dimension of balance of power or control. Collaborations with dominant partners, as for example occur in supplier networks for large manufacturers of cars or airplanes, are quite different from more peer-to-peer-oriented collaborations where no partner is dominant.

In terms of process management, it is convenient to make the distinction between service provision or consumption on the one hand and collaboration on the other hand. Starting from the precept that all processes have a single initiator, a process can be seen as collaborative if, at operational level (not purely on an intellectual basis with human owners), the initiator and executor have some awareness of each other's process and use this in some aspects of operations. The extend of this awareness (or use) of the other processes can be limited, for example to monitoring, but there should be some concrete use of the fact that the collaborator process exists.

While processes can and do exist independently of supporting (software) systems, this paper focuses on automated processes that are managed through some form of workflow or business process management system. In this digital representation, a process is represented as control flow as well as activities whose interaction is managed in line with the control flow to provide the process outcomes. Automation here implies that the process coordination is automated, not that all activities within them are automated. In some ways, human agents still have important roles to play in enactment and decision making.

In traditional centralized process management systems process, choreography is seen as performed by a single process management system (or process engine). This system has a full view of the process and all components. In case that external components are involved, they are treated opaquely as service consumption where the system itself has no further knowledge of the underlying processes or possible interactions. Of course it is, however, still possible to control interactions between multiple external services even with the same (external) owner.

In such a scenario, collaborative process management would imply that the management system requires quite some insight into, and control over the implementation of activities within the collaborating organizations. While cloud com-

puting makes this insight somewhat easier the main issue remains that of confidentiality and control. As such, many collaborators may object to this. In most cases, tight contractual arrangements are needed to allow this collaboration to happen [27,33], and this is mainly seen in a context with dominant players.

3 Failure modes of collaborative process systems

The various process system aspects can fail in various ways, but they share many aspects of the ways in which they can fail [35]. For the purpose of limiting the scope of the work, and avoiding the circular reasoning of making a robust system robust against errors in its own design or implementation, we look at failure sources that are out of direct control of the process system designer. The implementation (in contrast to the design) of a robust process management system is out of scope. Instead the focus is on managing causes of failure external to the system implementation in areas such as the behavior by external parties, equipment failure and other external incidents (e.g., fires, power failures).

In addition, we are also looking at failure from a software perspective, where manufacturing failure modes inform the perspective taken, but are not sufficient to analyze interactions across levels of resilience or the kinds of systems. Overall we distinguish the following orthogonal dimensions (please note that these are not exclusive):

- *Failure temporality* Along the time dimension, a failure can be temporary (an independent temporary event, for example a network failure due to a power cut); failures can be intermittent based on a systematic issue (related failures for example due to insufficient capacity during nightly batch operations); failures can also be permanent (the company providing the service has seized operation).
- *Failure detectability* Some failures are easy to recognize such as those where error messages are provided, signaled failure. Other failures are detectable based on business rules or SLA's such as failure to respond within a certain time period, detectable failure. The final category of failure detection is silent failures either in *silent non-performance* (i.e., the request is accepted but never actioned), *silent partial-performance* (i.e., some aspects of the request are not performed, but those aspects performed are valid). The final form of silent failure is silent corruption where the system provides incorrect state or results without signaling any error.
- *Failure completeness* Failure completeness concerns the degree to which the system is failing. Full failure is a possibility, but partial failure can occur in various forms such as degraded performance (e.g., not meeting time expectations), reduced quality (the results are valid but not of the quality or precision normally provided, the range of the function is reduced) or reduced provision (the domain or image of the functionality is reduced, i.e., not all inputs are supported, for example a weather service would have reduced provision for an area if the weather station in that area was unavailable for some reason).
- *Failure origin* The origin of a failure can either be *internal* as in it is caused by issues within the logic control of a system, or *external* where the failure is outside of the control of the system. An internal failure could be a bug in a service implementation or process description. A bug in the process management system would be external from the perspective of a process. A network failure that causes a service to be unavailable would be external to that service. Both examples can, in circumstances, be seen as internal to the overall system. An internal failure can either be a *delegate failure* (a used component causes the problem) or a *direct failure* where the problem is not fundamentally in the delegate (although a delegate may return unexpected (but not invalid) results that trigger the issue).

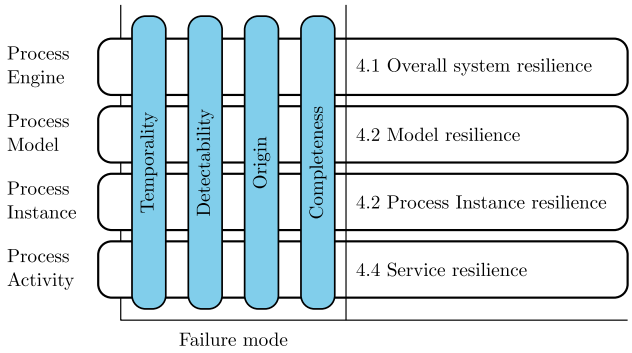

Fig. 1 Process resilience structure

Resilience is a measure of a system's ability to bounce back from a failure to continue to offer some level of performance. Understanding related failure models of collaborative process systems helps to decompose the measurement of resilience of the system. Overall this leads to a structure as depicted in Fig. 1, including analysis of the resilience on those dimensions in the relevant parts of Sect. 4.

4 Resilience of SOA collaborative process systems

Resilience and failure are closely related concepts. The overall description of a resilience model of an SOA collaborative process system is a combination of different resilience factors of different aspects of the system.

Francis and Bekera [15] have proposed a resilience analysis framework and a metric for measuring the resilience of an electric power network. This section discusses how, based on resilience factors, this framework can be extended to SOA collaborative business process systems. The different resilience factors are defined based on different perspectives of the system layers. As such, the proposed resilience model is based upon the combination of the analysis of the different constituent components of the SOA collaborative business process system.

In the context of this work, resilience can be defined as a combination of two aspects. The first aspect is the ability of the system to contain and minimize the effect of any disruption. The second aspect is the recovery profile of the system; how does the system reduce the impact of disturbances over time including the use of temporary fallbacks (e.g., using a backup server) until the resumption of normal operations (e.g., primary and backup systems are functioning as expected). In this context, a disruption is an event, such as a fire that leads to failures of the processes coordinated by the process management system. Another way of looking at this perspective of resilience is as the impact of a disruption (or failure) over time.

4.1 Resilience from an overall system perspective

As a consequence of looking at resilience as impact over time (modeling the impact as an abstract system performance level), it is possible to conceptualize an impact graph with performance on one axis and time on the other (see Fig. 2). On the temporal axis, various recovery stages can be distinguished as well as an overall time to recovery. As such, we define F_o the original stable system performance level, F_d the performance level immediately post-disruption once equilibrium state has been achieved; $F_r{}^*$ be the performance level after an initial post-disruption equilibrium state has been achieved; and F_r be the performance at a new stable level after recovery efforts have been exhausted. t_δ represent the duration of disruption.

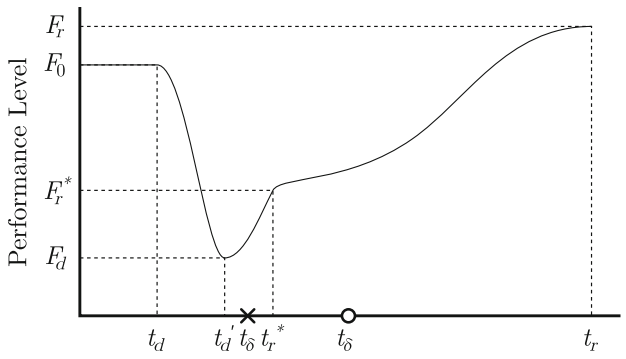

Fig. 2 System performance after initial disruption. Adopted from [15]

To model this, we define S_p to be the speed recovery factor. Under the disruption event i, a resilience factor ρ_i:

$$\rho_i(S_p, F_r, F_d, F_o) = S_p \frac{F_r}{F_o} \frac{F_d}{F_o},$$

where

$$S_p = \begin{cases} (t_\delta / t_r{}^*)\exp[-a(t_r - t_r{}^*)] & \text{for } t_r \geq t_r{}^* \\ t_\delta / t_r{}^* & \text{otherwise} \end{cases}$$

t_d = start of the disruption.
t_d' = equilibrium point at which the disruption causes no further performance degradation.
t_δ = slack time, i.e., maximum amount of time to post-disruption equilibrium that is acceptable before ensuring recovery.
t_r = time to final recovery, i.e., new equilibrium state.
$t_r{}^*$ = time to complete initial recovery actions.
a = parameter controlling decay in resilience attributable to time to new equilibrium.

Normally, when an interruption occurs, some initial actions are taken to stabilize the system at some intermediate stage after a disruption. Time to recovery, on other hand, is the length of time post-disruption until a system is brought back to reliable and sustainable performance in the long term. The resilience factor explicitly incorporates the time to recovery by comparing the time required for initial actions to be completed to a slack time for post-event recovery, while incorporating a decay factor to account for increases in the time it takes to reach the final post-disruption state. Figure 2 illustrates related concepts.

If the initial recovery takes longer than the slack time ($t_r{}^* > t_\delta$—illustrated by the cross in Fig. 2) then the resilience metric decreases (i.e., $1 > t_\delta / t_r{}^*$). If the initial recovery is quite efficient ($t_\delta > t_r{}^*$—the position on the horizontal axis marked by a circle), but the system takes a long time to recover after initial stabilization actions, the resilience metric also decreases.

Additionally, the hardness of the system in terms of the proportion of original system functionality (performance) retained immediately post-disruption is F_d / F_o. The proportion of original system performance retained after the new stable performance level has been achieved, as F_r / F_o. Notice that if $F_r > F_o$, it means that the system performance could be improved after the disruption. In general, the more functionality is retained relative to original capacity, both after an initial post-disruption and after a final recovery, the higher the resilience.

To better understand these ratios, consider a case where invoked services are performing poorly due to the accessibility or degraded QoS of network at certain areas. F_d / F_o shows

to the proportion of normal service level maintained despite the disruption. Suppose that the network or service provider has a temporary solution to the service of networks or services in some areas and, in the meantime, new version of services or/and recover efforts aimed at restoring everything to bring a new initial performance level $F_r{}^*$. Depending on the damage level, the recovery process may take up to more effort. Eventually, the collaborative system may be able to restore most of its services thereby achieving a new equilibrium. Hence, $F_r > F_o$ would mean the proportion of the normal system performance retained at the new equilibrium.

Let μ is defined as the probability of system failure. Let $f(\cdot)$ be the probability density function for system failure. Suppose further that system failure is a function of a parameter vector z. The possibility of the system failure under event i is

$$f(\mu|z)$$

The combination of the possibility of the system failure and resilience factor is as follows:

$$f(\mu|z)|_{t_d,K}$$

The possibility of occurrence of the event D_i in the system level, a derivation of a measure of expected system functionality degradation is defined as follows:

$$\xi = \sum_i Pr[D_i] \cdot f(\mu|z) \cdot \rho_i(S_p, F_r, F_d, F_o)$$

4.2 Resilience of individual process model perspective

In general, effective business processes must be able to accommodate changes in the environment [41]. Within imperative process modeling, parallelism contained at design time includes the actual ordering of activities open and thus provides more flexibility than sequential routing [43]. Some process models support 'deferred choices' which leave the resolution of a choice to the environment at runtime [43], YAWL [42] employs a *worklets approach* [1] to allow for late modeling. The late modeling allow a new process fragment to be constructed from scratch or composed from existing process fragments in order to complete a node which is marked as underspecified. The resilience of an individual process also depends on whether there is case handling and exception handling parts for the process model. Declarative process modeling is easier to defer choices to runtime [30,45].

If the individual processes consider and handle all error conditions (events, cases, exceptions, etc.), the resilience factor related to the individual process is defined as $\rho_i = 1$ under the condition/case/exception i. If the individual processes are designed with insufficient consideration of flexibility, the resilience factors related to the individual process will be $\rho_i = 0$ under the condition/case/exception i. A resilience factor ρ_i for an individual process is defined as:

$$\rho_i = \begin{cases} 1 & \text{specified to handle conditions, cases, exceptions} \\ 0 & \text{otherwise} \end{cases}$$

If a process model is widely used in the system, the design of process models could impact the system resilience. The more flexible process models designed at the design time, the stronger the resilience of the system could be.

4.3 Resilience of individual process instances perspective

Some business process management systems [37,44] allow process instances to deviate at runtime from the execution path prescribed by the original process without altering the process definition itself [38,41,46]. The deviation can only complete changes to the execution sequence for a specific process instance and does not need to modify original process models.

Another example of change at the process instance level is a momentary change [36,41], which is a change affecting the execution of one or more selected process instances only. An example of a momentary change is the postponement of registering a patient that has arrived to the hospital emergency center: treatment is started immediately. Such a momentary change performed on a given process instance does not affect any future instances.

In short, the above-described cases or events are able to be handled by the related systems with specific business process modeling languages, the resilience factor of the individual process instance $\boldsymbol{\rho}_i = 1$.

However, not all process instance failures are can be prevented in above-mentioned systems. There are different failure modes related to process instance failures, i.e., temporal failures, detectable failures, partial failures that cannot be handled at process model and instance level. There are also combinations of different failures for which specifying solutions are likely infeasible.

From the temporal perspective, some process instances are, due to their inherent long-running nature, reasonably resilient to non-permanent failures. The lower the pressure on time, the stronger the resilience could be.

Detectability is another perspective of analyzing the resilience of an SOA collaborative process systems. Silent non-performance and some silent partial-performance process instances can possibly be handled through monitoring and timeouts, which reflect to time $[t_d, t_d{}']$ in Fig. 2.

Therefore, the resilience factors of process instance are defined as

$$\rho_i = \begin{cases} 1 & \text{specified conditions, cases, exceptions} \\ S_p \frac{F_r}{F_o} \frac{F_d}{F_o} & \text{otherwise} \end{cases}$$

At process instance level, factors (e.g., events, conditions) whose handling was not specified in the process model could impact any number of process instances. The total count impacted process instances need to be analyzed under the occurrences of unspecific events. The impact process instances could be based on a same process model and could also be based on related process models. Measuring the impact of correlative process instances based on different process models is needed to reflect real-world process enactment, but comes with additional challenges.

To increase resilience of the system, possibility of the individual unspecified factors of the process models needs to be analyzed and collected. Whether the handling of the unspecified factors should be specified in the process models or should be left to the process execution engine influences the improvement in the system resilience, in particular due to the differing amount of effort involved. Designing preventive or proactive process models and runtime case handle mechanism is crucial for the system resilience improvement.

4.4 Resilience from a service perspective

Services form the building blocks of a SOA-based process system. From the service perspective of this system, the external services are provided by external systems and cannot be directly managed. The causes of their failures are therefore out of scope of this paper, but they may exhibit all forms of failure modes, i.e., temporal failures, silent non-performance or partial-performance failures. These constituent service failures do impact the overarching resilience levels of the system. Overall, the system is more resilient when it is easy to (automatically) replace different similar servers from the same providers or different services from the different providers for the process (activity execution or process management).

To increase resilience of the process instances, it is important to identify the specific services which are commonly involved and conditions of service replacement for these services. Although, even when services cannot be readily replaced, service failure won't always lead to instance failure, it is still important to know the following:

- given the 100% failure of a specific service to further identify what percentage of the process instances will fail;
- given the 100% failure of a specific service, what percentage of all instances will fail.
- the maximum percentage of impacted instances—all instances that may fail due to service failure, however, unlikely
- the minimum percentage of instances—all instances that will certainly fail due to service failure

These specific high impact services should be identified. To improve overall system resilience, it is important to have an automatic mechanism of replacing the services with the services from the same providers (in different locations) or different services from the different providers for the process (activity execution or process management).

4.5 Resilience from a resource perspective

Elastic service provision [21] in a cloud-based environment is a novel approach executing processes using the resource elasticity of cloud-based systems. In such a cloud-based environment, elastic services can be hosted whose resources may dynamically grow and shrink to meet the requirements of dynamically varying numbers of users and patterns of requests.

The execution of individual process instances is performed by a resource, whether it is a server, a single human (through a worklist manager) or a pool of resources. The resilience of the collaborative system increases by elastically accessing to a pool of resources, e.g., computing power with auto-scaling, services with auto-replacement, extra human resources.

5 Measuring resilience: a collaborative notification case

5.1 Collaborative incident and maintenance notification case

The Spanish electricity system is generally formed of a high-voltage electric power transmission network and grid connecting power stations and substations to transport electricity from where it is generated to where it is needed.

There are a number of stakeholders in the Spanish electricity system, each fulfilling various roles in the overall process of electricity generation and delivery. Many of the well-established, former government, parties play many of the roles in the system. The unavoidably monopolistic and critical roles of market operator (OMEL) and system operator (REE) remain in government hands. There are three main energy producers, one market operator, three main distributors, eighteen substations, and 32 marketers.

The process of delivering electricity to a single customer involves the entire chain of roles (and therefore actors). The delivery of electricity to a single geographic group (for example, a street) of customers likely involves many

more marketers. Given that most distributors are also active as marketers on a national level, these distributors generally act both as each other's collaborators/customers and competitors.

Incidents in the electricity system can occur anywhere and anytime. These incidents, ranging from signal errors, cabling problems to serious substation overloads, will affect energy supply, lead to power cuts or even generate a further huge impact to the community and economy. Maintenance activities are regularly taking places in different organizations within the electricity system. These maintenance activities increase the risk of failures and may trigger trigger interruptions within the electricity systems.

Large commercial customers directly connected to the 132 kV network are generally on interruptible contracts. As their electricity demands can have significant impact on the network there is frequent and well-established contact with these customers that could be used for incident notification. Residential customers and smaller business users, however, are often only known by name to marketers. The distributors and market operator generally only have knowledge of the addresses affected by a power cut, but no further contact details. As a result, residential and small business users are currently not notified of incidents directly (for larger disruptions they may be informed indirectly through the local press).

To improve customer satisfaction, it becomes clear that effective incident management includes effective and timely informing of all customers without relying on suddenly overloaded call centers. The information provided should not only acknowledge the existence of an issue but also provide information on progress and estimated resolution timelines. Appropriate, follow-up notifications should be sent to all or interested customers.

5.2 Cloud-based SOA collaborative process solution

We propose a cloud-based SOA collaborative process solutions for incident and maintenance notification for the Spanish electricity system [48,51] as an illustrative case for determining resilience. The incident notification process used as our case is based upon the use of business process-oriented mashup engines. These business process-oriented mashup engines are deployed for all distributors and marketers. This insures that all involved stakeholders can flexibly deal with appeared incidents (the engines can also be used for other situational applications).

The architecture of the cloud-based incident and maintenance notification process solution (CIMNPS) has a number of subsystems. The user management subsystem provides access control for all stakeholders. The process management subsystem handles process uploading, process editing, process ranking and selection, as well as service discovery. The service runtime management subsystems handle monitoring, reporting and service invocation at the runtime. Furthermore the CIMNPS also has two repositories for collaborative processes and services related to incident notification.

Business process-oriented mashup engines are deployed for all stakeholders. The business process model repository contains collaborative business process models. This repository is managed by the overall system owner OMEL and enables sharing and reusing of existing collaborative process models. The business process editor allows modifying, verifying, and ranking process models. Each stakeholder has access to the process editor and is able to make changes to its processes as long as that does not compromise the integrity of the overall system. The collaborative process models can be downloaded or uploaded to the business process model repository as desired. Process models are instantiated into process instances after all data sources and invoked services are (semi-) automatically identified. Figure 3 presents a collaborative notification process model, which could help to understand overall functions of the CIMNPS.

The decentralized execution of process instances is a core aspect of the incident notification system in the Spanish electricity system case. While the information needed from both distributors and marketers is not large in terms of data size (so transfer would not be a technical challenge), there are data sensitivity issues. While the information from the distributors is not commercially sensitive, the data from the marketers is. For the purpose of incident notification, marketers could download a common process model from the process model repository on the cloud. Starting with the downloaded process model, the marketers can make modifications; for example to adopt different notification channels, such as sending Facebook messages as well as SMS messages. The marketers can use a local process editor to allocate the data (affected customers' mobile phone number, Facebook ID, or Twitter ID) and run the business process on a private process-oriented mashup engine. For the distributors, incidents or interruptions can be caused or observed in different parts of the organization. Therefore, a sample process model can be modified according to the situation. The process can keep monitoring the process of the repair and ensure information is consistently published on the web using the private process engine.

The solution owner (in this case OMEL) mainly concentrates on maintaining the process model repository and on providing some common web services. OMEL also provides a process editor and a process engine for users testing the process model. The other users are certainly able to upload their data for running their processes in case the private process engine is out of order. A collaborative process which runs in the cloud can be supported for special cases, e.g., monitoring the collaborative process.

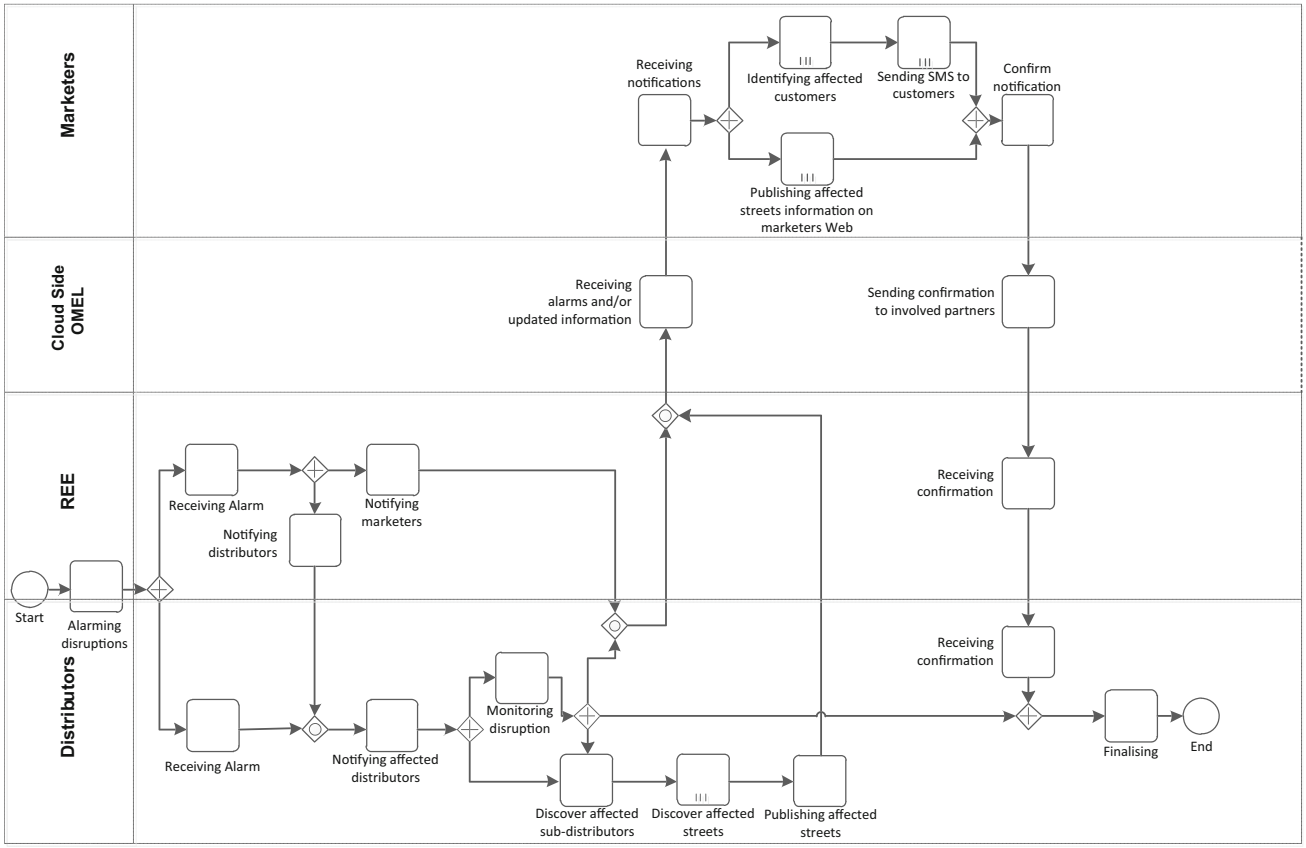

Fig. 3 Cloud-based incident and maintenance notification process model

5.3 Resilience analysis of CIMNPS

In Sect. 5.1, we presented the case of the Spanish electricity system. The major stakeholders are summarized in Table 1. In regards to these stakeholders, there are four major incident categories, i.e., incidents related to main grid, incidents related to main distributors, incidents related to main energy producers, and incidents related to substations. The incidents could also be combinations of the above-mentioned categories. The major maintenance categories are double of the incident categories (i.e., eight major maintenance categories), which one kind of maintenance could be internal or external. Therefore, there are at least 12 collaborative process models for supporting the incident and maintenance notification in the CIMNPS. Figure 3 presents only one high-level collaborative notification process model in CIMNPS.

Furthermore, there are also local process models, which exist within one stakeholder organization for notification purposes. There are thus possible 56 local process model variables. The total kinds of notifications could be incident notifications, different maintenance notifications; maintenance caused an incident notification and further combines internal and/or external notifications.

Table 1 Main stakeholders in the Spanish electricity system

Main stakeholders	Number
Main energy producers	3
Main distributors	3
Substations	18
Marketers	32

5.4 Experimental set up

We set up scenarios as there were energy cuts of between 6 to 60 h. 100,000 households are affected in a big city. After 6 h, initial recovery actions are done. After 60 h, the incident was solved, no message needs to send. The process of sending incident notification to residential consumers is a highly collaborative process. Each involved partner needs to provide data collaboratively.

Normally all marketers and other participated organizations run their data individually based. The marketers could decide to send their customers' basic contact information to the cloud owner OMEL for distributing all notifications about incident recovery. The marketers could also send incident notifications to their customs after the OMEL announcement by themselves.

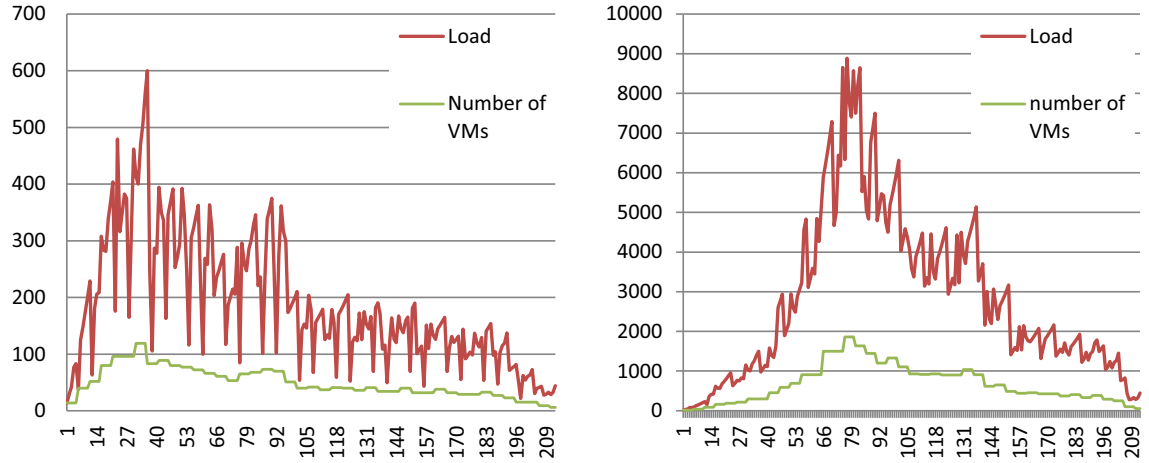

Fig. 4 Computing power costs of two hypothetical workload scenarios

To run the experiment, we distinguish two Internet network bandwidths: one is among involved enterprises; and another is between marketers and the cloud. Depends on where the residential customers' data (i.e., at the marketers' sides or, centralizing it at OMEL side) is, the speeds of sending out notifications are thus different.

We select two hypothetical workload scenarios. The first workload scenario represents an incident or maintenance event that initially impacts a large amount of households. After an initial buildup time, temporary resolutions are quickly put in place such that many households are no longer impacted such that further notifications are no longer required for these households. The notification process is completed after the incident has been fully resolved. An example of such an incident could be a broken cable where neighborhoods and other larger groupings of households are gradually rerouted well before the broken cable is repaired.

The second workload scenario depicts an incident or maintenance event that escalates before slowly being resolved. Initially few households are impacted, but this amount grows significantly before the issues are eventually resolved gradually. The incident is eventually resolved after some time.

5.4.1 Resource-level resilience analysis

There are many different resources which could be used for running a system. The resources of the CIMNP systems include computing powers, process engines, human resources.

One of the considerations in the design of these systems is whether using elastic cloud resources is appropriate and worthwhile. In Fig. 4, the dashed line represents the needed computing power without auto-scaling virtual machines (VRs) and the solid line denotes the computing power with auto-scaling VRs involved in computing the power costs of the two hypothetical workload scenarios as mentioned in Sect. 5.4.

Table 2 Resilience factors of computing powers

	Resilience factors ρ	
	No VRs	Using VRs
Scenario 1	1	1.42753
Scenario 2	1	1.417

In this experiment, the scaling policies are set as: if the average CPU load is below 40% for 5 min, the number of virtual machine is reduced; if the average CPU load is above 70% for 5 min, one new virtual machine is added [18].

The examples of the resilience factors for the above-mentioned computing power resilience (ρ) are presented in Table 2, Scenario 1 refers the hypothetical workload scenario, which an incident initially impacts a large amount of households. In this scenario, to process the large amount of requests five VRs are used. Using one VR, there is 30% computing power increase for the system. Therefore, ρ is [1, 1.42753].

Scenario 2 refers to the hypothetical workload scenario: an incident/maintenance event that escalates before slowly being resolved. In this scenario, three VRs are used; therefore, ρ is [1, 1.417].

The design of CIMNP deploys that multiple process mashup engines run at different partners. This allows them to function as backups for each other, i.e., at different situations. Resource-level resilience relates to elastic computing powers, redundancy of process engines, data redundancy.

5.4.2 Service-level resilience analysis

Services are building blocks for the CIMNPs. Similar to processing capacity, (web) services may not be available or over

their capacity (possibly caused externally to the system). Automatic service replacement or service invocation reallocation mechanisms need to be designed at the system level. In cloud environments many services may transparently scale on the demand, or allow for the specification of an alternative instance to invoke.

Service-level resilience could also be improved by predesign of service invocation reallocation mechanisms and use of scalable services on demand. The consideration of such mechanisms should be kept in mind at the design and execution configuration stages.

5.4.3 Process incident-level resilience analysis

Process instance resilience relates to how process instances can recover from failures. In this aspect, direct failures are normally due to incorrect assumptions or pure logic errors (bugs in the process model), and of less interest within the resilience discussion.

From a temporal dimension perspective, process instances are, due to their inherent nature as long-running, reasonably resilient to non-permanent failures. The lower the pressure on timing, the stronger the resilience.

From a detectability perspective, only signaled failures are normally handled by process instances, and higher resilience generally implies more complete failure management, that includes the availability of sufficient semantic information such as to allow automatic restarts or rerouting for failed paths/components. Silent non-performance can possibly be handled through monitoring and timeouts, but in many cases cannot be distinguished from other forms of silent failure.

In terms of partial failure, degraded performance may lead to timing issues on the process instance, but otherwise should not impact the performance of the process. Reduced quality of performance leads to reduced quality of the instance, and reduced provision would lead to some instances failing (those affected by the reduction) and some succeeding.

In relation to the location of failure, external failure is normally failure of delegates/activities, but can also be a failure in the execution environment (e.g., hardware failure). External failure of the execution environment could be compensated by regular replication solutions for the state of the execution environment (e.g., database replication/sharing) that is not specific to process management. In terms of direct failure, there are two aspects to the resilience of process models. First is the ability of humans to interject in process instances and possibly modify them or provide "missing" data; this is mainly a property of the management system and the used process language. The other aspect is the frequency of instances. In other words, while it may be technically feasible to intervene in a process instance and recover it, it may not be feasible for large amounts of instances that failed due to fundamentally the same cause.

For CIMNPS, an incident and many maintenance notification process instances could be running concurrently. For example, this is the case in the first scenario we mentioned in Sect. 5.4. When a number of maintenance notification processes are running, an incident impacts a large amount of households. The running process instances need to be prioritized for overall reduction in the effect on system performance. Such prioritization mechanisms should be included at the design stage to improve the resilience of the system.

5.4.4 Process model-level resilience analysis

The resilience of processes is strongly related to the resilience of process instances. There are, however, ways in which the execution system can make processes be resilient even when individual instances fail. Conversely, timing and speed issues that are not problematic at instance level, may become so at process level.

Process resilience at a temporal level requires monitoring and (pre-emptive) intervention and/or rerouting. Processes are resilient when it is easy to (automatically) replace the providers of services for the process (activity execution or process management), but it requires a resilient system to actually make use of those possibilities. Obviously if these measures can be performed on already started process instances, this would even be better.

At process level, the detectability of failures is limited to the ability to express overall expectations, possibly at aggregate level. These process-level specifications could then allow for pre-emptive intervention in the instantiation of new processes. Note that at process or system level it may also be feasible to determine some degree of likelihood of the various forms of silent failure by aggregating monitoring data and linking it to the expectations.

At process-level partial failure of some instances would constitute partial failure of the process. A resilient process would allow the instance failures to be minimized and possibly corrected through re-execution along an alternative path. To do the latter, does require sufficient information to be available for the process components to be able to determine the semantic validity of doing so.

Locus of failure at process level implies some degree of process quality requirements as well as designs that minimize the impact of external failures. Again, monitoring of individual service and instance performance can help in detecting, mitigating and avoiding these failures or transitioning them from non-performance to degraded performance by for example choosing a lower quality service that does not have availability issues.

As mentioned in Sect. 4.2, selecting right process modeling languages and process running environments is very important for improving the process-level resilience. The

process running environment is even more important, which decides the resilience of the system.

5.4.5 System-level resilience analysis

Overall the entire systems resilience is a combination of the resilience of the constituting processes. Many measures at overall level can, however, help to achieve this resilience. In particular the provision of fallback execution environments or services will improve the overall resilience of the system. Resilience in terms of fault tolerance mainly depends on the resilience of the components (services, processes) and sufficient provision of execution environments without single point of failure.

The previous section discussed the resilience of the components that make up CIMNPS system on a technical level rather than at stakeholder level. Organizational factors may limit options or have external impacts on system design and resilience. The possibilities of collaborative process management can, on a theoretical level be seen as a subset of what single entity-controlled process management provides. On a practical level, there are resilience benefits to collaborative process execution (there are also disadvantages in complexity and management). In addition, in real-world business scenarios it is often not desirable to have centralized process control.

In terms of resilience, collaborative process management means that by implication, processes are managed by multiple independent execution environments. While these environments may have access to different components (such as company internal services) they would have similar orchestration capabilities. As collaborative processes allow parts of execution to be delegated to other execution environments this means that an overall master coordinator does not require direct access to all required components for process execution.

At system and process level, through a lack of single point of failure and lack of a global coordination requirement, cloud-based SOA collaboration solution provides a strong level of resilience against total failure. Performance degradation, especially non-functional degradation, can be compensated for by additional provisioning elsewhere within the collaboration. Even the absence of certain unique resources, would only result in those processes being unavailable.

The services used by a system are independent of the form of process management, and their resilience is mainly a property that influences higher levels. Distributed collaboration, however, brings some interesting perspectives to process instance resilience.

Within hierarchical distributed collaboration it is possible that, beside the master execution environment multiple delegate environments are active concurrently. Fallback, if semantically valid within the process, would, however, still be possible as long as a strict order of fallbacks is known by all candidates. These candidates would also be required to have some degree of overall knowledge of the process, possibly under some sort of escrow approach.

6 Related work

The notion of resilience has been getting attention for many different disciplines. Resilience is an emergent property associated with an enterprise system's capability to continue its mission despite disruption through. We reviews resilience related work from overall conceptual definition and framework of resilience analysis from ICT systems, enterprise information systems, business process management systems, design principles of enterprise systems, to resilience analysis and simulation model for SOA enterprise systems.

6.1 Design framework and principles of ICT systems

Di Marzo Serugendo et. al. [11] present a metadata-based architectural model for dynamically resilient ICT systems, which support different sources of data from communication providers such as GPS, sensors. The design considers predictable dynamic resilience of open systems built from components that interact via a network-based infrastructure. The proposed architecture is thus emphasis on to deliver dynamic resilience within bounds that can be predicted at design time.

Papers [52] and [28] provide earlier work on design of resilient enterprise information systems. Different resilience concepts from different disciplines are reviewed. Different capabilities or concepts related to resilience are distinguished. A guideline for further developing and implementing practical enterprise systems to be more resilient are discussed in [52]. Paper [28] further presents seven architectural constraints for resilience. The seven constraints are elicited from good architectural practices for developing reliable and fault-tolerant systems and the state-of-the-art technologies in distributed computing. Again, the provided design principle of enterprise systems is in an abstract level for guideline proposes.

Paper [14] proposes a framework to create enterprise resilience using service-oriented architecture approach. The proposed framework provides an abstract theoretical model, which explains how SOA can help in creating enterprise resilience. No quantitative analysis of resilience of SOA systems is provided.

Müller et al. highlight importance of resilience for business organizations [32]. The paper explicitly addresses the different capabilities of an organization being resilience and explains how to incorporated resilience into existing business information system design in an abstract level.

Antunes and Mourão consider a framework and related services for building business process management (BPM) to be more resilient [4]. This work is more specific for designing business process management systems which is one kind of enterprise systems. The paper looks the resilience related supports in the current business process management systems, i.e., failure handling, exception handling, model adaptation, restricted ad hoc changes, and unstructured interventions. Furthermore, a set of services integrating resilience support in BPM system are discussed,

6.2 Resilience analysis and simulation mode

Jassbi et.al proposes a framework for evaluating resilience of disaster rescue networks [19]. The work aims to provide an evaluation mode through employing experts' knowledge, for measuring a resilience index in disaster rescue networks. A quantitative mode is presented to keep a record of rescue networks and how they are achieving adequate levels of resilience, which provides a resilience simulation model.

This research is based on our previous papers [10] and [9] from PROVE 215 and PROVE 216, respectively. Paper [10] presents an approach to analyze resilience of a SOA collaborative process system in cloud-environment. The resilience of the solution is related to a service-level agreement (SLA) of the maximum time to process certain services against the actual time to process certain services. The method is adapted from the work of Marcon et al. [29], which provides resilience to the re-evaluation of usage policies of individual users. The resilience is providing the model with the ability to deal with some individual user authorization attributes exceeding, which the SLA for the respective consumption service is under the contracted amount. Our another previous work [9] reviews federated collaborative process management and analyses the resilience of federated collaborative process management in the new context.

Francis and Bekera propose a resilience analysis framework and a metric for measuring resilience [15]. The framework is focused on the achievement of resilience capacities for engineered and infrastructure systems (e.g., electric power networks). Their resilience framework consisting of attributes such as adaptive capacity, absorptive capacity, and recoverability enables analysis of system performance for highly uncertain or unforeseeable environment. Based on their work of resilience factor, we extend to the SOA collaborative business process system. Different resilience factors are defined based on different perspectives of the system layers. Such SOA collaborative business process systems could be designed as BPaaS solutions with elastic services power and deployed in cloud-base environments. Our work presented in this paper addresses these new trends for analyzing the resilience of SOA collaborative business process systems.

7 Conclusions

From the discussion above, it is clear that beyond meeting business needs in providing the benefits of collaboration while safeguarding independence of control, collaborative process environments also bring interesting resilience properties. Improving the resilience of SOA-based collaborative systems is quite feasible when it is easy to (automatically) replace the providers of services for the process (activity execution or process management), but it requires a resilient system to actually make use of those possibilities.

The fundamental challenges of providing an appropriate definition of resilience of SOA collaborative process management system, analyzing, measuring, and further improving the resilience of these systems have motivated our research. We thus look at the different perspectives of the SOA collaborative process management systems, respectively. From the *resource- and service-level* perspectives, elastic computing provision and services with automatic replacements are certainly able to handle unpredicted workloads. At *process model level*, parallelism contained at design time includes the more flexible order of execution of activities. This means that the selection of process modeling languages and execution environments at design time is important. Process modeling languages supporting deferred choices could leave the resolution of a choice to the process engine at runtime, which provides flexibility for individual process instances. At the *system level*, the running process instances need to be prioritized for overall reduction in the effect on system performance. Such prioritization mechanisms need to be included in the design to improve the resilience of the system. Continually monitoring and proactive model process executions and collecting unspecified factors and analyzing their impacts to the system feedback to the refine the system and improve the case handle mechanism at runtime, which is essential for improving system resilience.

The resilience analysis of CIMNPS has demonstrated how to determine resilience factors of different perspectives within CIMNPS. The resilience factor at the resource level of CIMNPS is analyzed. The potential resilience improvements have been discussed at service level, process instance level, process model level, and system level, respectively. Furthermore, we look at related work around resilience analysis of ICT systems.

There are some limitations to our current research. *Service-level* and *process incident-level* resilience analyses, needed for analyzing the resilience of CIMNPS, are not included in this paper. More running data need to be collected for the analysis. The further analysis of the *overall system level* is needed to identify failure events, performance degradation, etc., which will allow us to analyze the expected system functionality degradation (ξ).

Acknowledgements This research has been partially sponsored by EU H2020 FIRST project, Grant No. 734599

References

1. Adams M, ter Hofstede AHM, van der Aalst WMP, Edmond D (2007) Dynamic, extensible and context-aware exception handling for workflows. In: Meersman R, Tari Z (eds) On the move to meaningful internet systems 2007: CoopIS, DOA, ODBASE, GADA, and IS. OTM 2007. Lecture notes in computer science, vol 4803. Springer, Berlin, pp 95–112. https://doi.org/10.1007/978-3-540-76848-7_8
2. Afuah A (2001) Dynamic boundaries of the firm: are firms better off being vertically integrated in the face of a technological change? Acad Manag J 44(6):1211–1228
3. ALICE (2014) Information systems for interconnected logistics research and innovation roadmap. http://www.etp-logistics.eu/?page_id=89. Accessed 5 May 2017
4. Antunes P, Mourão H (2011) Resilient business process management: framework and services. Expert Syst Appl 38(2):1241–1254
5. Camarinha-Matos LM (2014) Collaborative networks: a mechanism for enterprise agility and resilience. In: Mertins K, Bénaben F, Poler R, Bourrières JP (eds) Enterprise interoperability VI. Proceedings of the I-ESA conferences, vol 7. Springer, Cham, pp 3–11
6. Camarinha-Matos LM, Afsarmanesh H (2005) Collaborative networks: a new scientific discipline. J Intell Manuf 16(4–5):439–452
7. Campbell FC (2008) Elements of metallurgy and engineering alloys. ASM International, Materials Park. ISBN 978-0-87170-867-0
8. Christopher M, Peck H (2004) Building the resilient supply chain. Int J Logist Manag 15(2):1–14
9. de Vrieze P, Xu L (2016) Resilience analysis of collaborative process management systems. Springer, Cham, pp 124–133. https://doi.org/10.1007/978-3-319-45390-3_11
10. de Vrieze P, Xu L (2015) An analysis of resilience of a cloud based incident notification process. In: Risks and resilience of collaborative networks. Springer, pp 110–121
11. Di Marzo Serugendo G, Fitzgerald J, Romanovsky A, Guelfi N (2007) A metadata-based architectural model for dynamically resilient systems. In: Proceedings of the 2007 ACM symposium on applied computing. ACM, pp 566–572
12. Dong Q, Bradshaw K, Ferrese F, Bai L, Biswas S (2011) Cooperative federated multi-agent control of large-scale systems. In: ACTA control and applications conference
13. EFFRA (2013) Factories of the future 2020 roadmap. http://www.effra.eu/sites/default/files/factories_of_the_future_2020_roadmap.pdf. Accessed 5 May 2017
14. Erol O, Mansouri M, Sauser B (2009) A framework for enterprise resilience using service oriented architecture approach. In: 2009 3rd annual IEEE systems conference. IEEE, pp 127–132
15. Francis R, Bekera B (2014) A metric and frameworks for resilience analysis of engineered and infrastructure systems. Reliab Eng Syst Saf 121:90–103. https://doi.org/10.1016/j.ress.2013.07.004
16. Gereffi G, Humphrey J, Sturgeon T (2005) The governance of global value chains. Rev Int Political Econ 12(1):78–104
17. Gou H, Huang B, Liu W, Li X (2003) A framework for virtual enterprise operation management. Comput Ind 50(3):333–352
18. Janiesch C, Weber I, Kuhlenkamp J, Menzel M (2014) Optimizing the performance of automated business processes executed on virtualized infrastructure. In: 2014 47th Hawaii international conference on system sciences (HICSS). IEEE, pp 3818–3826
19. Jassbi J, Camarinha-Matos LM, Barata J (2015) A framework for evaluation of resilience of disaster rescue networks. In: Working conference on virtual enterprises. Springer, pp 146–158
20. Katila R, Mang PY (2003) Exploiting technological opportunities: the timing of collaborations. Res Policy 32(2):317–332
21. Konstanteli K, Cucinotta T, Psychas K, Varvarigou TA (2014) Elastic admission control for federated cloud services. IEEE Trans Cloud Comput 2(3):348–361
22. Koslowski TG, Longstaff PH, Vidal M, Grob T, et al (2012) Resilience analysis of the ICT ecosystem. In: 23rd European regional international telecommunication society conference, Vienna, pp 1–18
23. Kutvonen L, Ruokolainen T, Metso J (2008) Interoperability middleware for federated business services in web-pilarcos. In: Protogeros N (ed) Agent and web service technologies in virtual enterprises. IGI Global, Hershey, PA, pp 288–309. https://doi.org/10.4018/978-1-59904-648-8.ch017
24. Kutvonen L, Metso J, Ruokolainen T (2005) Inter-enterprise collaboration management in dynamic business networks. In: On the move to meaningful internet systems 2005: CoopIS, DOA, and ODBASE. Springer, pp 593–611
25. Lavie D (2006) The competitive advantage of interconnected firms: an extension of the resource-based view. Acad Manag Rev 31(3):638–658
26. Leymann F, Roller D, Schmidt MT (2002) Web services and business process management. IBM Syst J 41(2):198–211. https://doi.org/10.1147/sj.412.0198
27. Liu C, Li Q, Zhao X (2009) Challenges and opportunities in collaborative business process management: overview of recent advances and introduction to the special issue. Inf Syst Front 11(3):201–209
28. Liu D, Deters R, Zhang WJ (2010) Architectural design for resilience. Enterp Inf Syst 4(2):137–152
29. Marcon AL, Santin AO, Stihler M, Bachtold J (2014) A UCON_{ABC} resilient authorization evaluation for cloud computing. IEEE Trans Parallel Distrib Syst 25(2):457–467
30. Montali M, Pesic M, van der Aalst WM, Chesani F, Mello P, Storari S (2010) Declarative specification and verification of service choreographiess. ACM Trans Web (TWEB) 4(1):3
31. Muckstadt JA, Murray DH, Rappold JA, Collins DE (2001) Guidelines for collaborative supply chain system design and operation. Inf Syst Front 3(4):427–453
32. Müller G, Koslowski TG, Accorsi R (2013) Resilience-a new research field in business information systems? In: International conference on business information systems. Springer, pp 3–14
33. Niehaves B, Plattfaut R (2011) Collaborative business process management: status quo and quo vadis. Bus Process Manag J 17(3):384–402
34. Norta A, Grefen P (2007) Discovering meta-models for inter-organizational business process collaboration. Int J Coop Inf Syst 16(03n04):507–544
35. Pentti H, Atte H (2002) Failure mode and effects analysis of software-based automation systems. VTT Ind Syst STUK-YTO-TR 190:190
36. Pesic M, Schonenberg M, Sidorova N, van der Aalst WM (2007) Constraint-based workflow models: change made easy. In: OTM confederated international conferences on the move to meaningful internet systems. Springer, pp 77–94
37. Reichert M, Dadam P (1998) Adeptflex—supporting dynamic changes of workflows without losing control. J Intell Inf Syst 10(2):93–129

38. Rinderle S, Reichert M, Dadam P (2004) Correctness criteria for dynamic changes in workflow systems—a survey. Data Knowl Eng 50(1):9–34
39. Sia SK, Soh C, Weill P (2008) It governance in global enterprises: managing in Asia. In: ICIS 2008 proceedings, p 97
40. Sturgeon TJ (2002) Modular production networks: a new American model of industrial organization. Ind Corp Change 11(3):451–496
41. van der Aalst WMP (2013) Business process management: a comprehensive survey. ISRN Softw Eng 2013:507984-1–507984-37. https://doi.org/10.1155/2013/507984
42. Van Der Aalst WM, Ter Hofstede AH (2005) Yawl: yet another workflow language. Inf Syst 30(4):245–275
43. van Der Aalst WM, Ter Hofstede AH, Kiepuszewski B, Barros AP (2003) Workflow patterns. Distrib Parallel Databases 14(1):5–51
44. Van der Aalst WM, Weske M, Grünbauer D (2005) Case handling: a new paradigm for business process support. Data Knowl Eng 53(2):129–162
45. van Der Aalst WM, Pesic M, Schonenberg H (2009) Declarative workflows: balancing between flexibility and support. Comput Sci Res Dev 23(2):99–113
46. Weske M (2001) Formal foundation and conceptual design of dynamic adaptations in a workflow management system. In: Proceedings of the 34th annual Hawaii international conference on system sciences, 2001. IEEE, p 10
47. Xu LD (2011) Enterprise systems: state-of-the-art and future trends. IEEE Trans Ind Inform 7(4):630–640. https://doi.org/10.1109/TII.2011.2167156
48. Xu L, de Vrieze P, Jiang N (2015) Incident notification process as bpaas for electricity supply system. Int J Cloud Comput 2(4):33–44
49. Xu L, de Vrieze P, Phalp K, Jeary S, Liang P (2010) Lightweight process modeling for virtual enterprise process collaboration. In: PRO-VE, pp 501–508. https://doi.org/10.1007/978-3-642-15961-9_60
50. Xu L, Jeusfeld MA (2003) Pro-active monitoring of electronic contracts. In: Eder J, Missikoff M (eds) Advanced information systems engineering. CAiSE 2003. Lecture notes in computer science, vol 2681. Springer, Berlin, pp 584–600
51. Xu L, Vrieze PD, Jiang N (2013) Incident notification process as a service for electricity supply systems. In: Proceedings of the 2013 IEEE sixth international conference on cloud computing. IEEE Computer Society, pp 926–933
52. Zhang WJ, Lin Y (2010) On the principle of design of resilient systems-application to enterprise information systems. Enterp Inf Syst 4(2):99–110