






Effects of Concurrency in Complex Service Organizations: Evidence from Electronic Health Records

Brian T. Pentland¹  , Inkyu Kim¹, Quan Zhang¹, and Julie Ryan Wolf² 

¹ Michigan State University, East Lansing, MI, USA
Pentland@broad.msu.edu

² University of Rochester, Rochester, NY, USA

Abstract. We use Kremser and Blagoev's [1] role-routine ecology to theorize about the effects of concurrency in complex service organizations, such as outpatient medical clinics. In a typical clinic, teams of specialized individuals serve multiple clients at the same time. There can be concurrency within a patient visit (a technician may be preparing for a procedure while the doctor talks to the patient) and concurrency between patient visits (multiple patients being treated in the clinic). Using data from electronic health records, we estimate the effects of concurrency within and between patient visits on the duration of patient visits in a set of dermatology clinics. As expected, we find that concurrency within patient visits is associated with reduced duration, while concurrency between visits is associated with increased duration. We discuss the implication of these findings for process mining and discovery of process models in organizations where process instances are not independent.

Keywords: Organizational routines · Role-routine ecology · Concurrency · Electronic health records

1 Introduction

A hallmark of process mining has been the focus on concurrency within a process. Van der Aalst [2] places an emphasis on concurrency and objects as way to reveal the “true fabric of business processes” [3]. At the same time, one of the on-going challenges in process mining and process management concerns the execution of multiple, concurrent process instances [4, 5]. For example, in an outpatient medical clinic, there is almost always more than one patient in the clinic at a time. As a result, the patients in the clinic are competing for the time and attention of the clinical staff. There is concurrency within patient visits, but also between patient visits. Thus, the outpatient medical clinic is an interesting context for theorizing about organizations where process instances are not independent.

In this paper, we examine the effects of concurrency within and between process instances using the role-routine ecology, a new conceptual framework from Kremser and Blagoev [1]. In a role-routine ecology, work is organized by the competing needs of

the routines (e.g., treating a patient) and the roles (e.g., being a physician in a clinic). In organizations that have a complex role-routine ecology, where multiple roles are engaged in multiple routines at the same time, the workflow is emergent. By emergent, we mean that the “existence and nature” of the behavior “depend upon entities at a lower level, but the behavior is neither reducible to, nor predictable from, properties of entities found at the lower level” [6]. We argue that the nature of the role-routine ecology has implications for process mining and discovery of process models.

We begin by defining the role-routine ecology and the concept of concurrency within and between process instances. Then we analyze electronic health record (EHR) data from a set of outpatient dermatology clinics to demonstrate the effects of concurrency. Finally, we discuss the implications of these findings in terms of the role-routine ecology. In a complex role-routine ecology, workflow is an emergent product of competing priorities, which raises challenges for conventional process mining and for the prospects of more sophisticated models, such as Digital Twins of Organizations [7, 8].

2 Background

2.1 Role-Routine Ecology in Complex Service Organizations

Kremser and Blagoev [1] introduce the concept of a role-routine ecology as a way to analyze the competing priorities that govern the work processes in a consulting organization. On one hand, actions are prioritized according to the needs of the routine: the “repetitive, recognizable pattern[s] of interdependent actions, involving multiple actors” [9] that are oriented toward the accomplishment of a “day-to-day operational task” [10]. On the other hand, actions are prioritized based on the needs of the role. Kremser and Blagoev [1] argue that “a role performance is a sequence of actions...”. Like routines, roles are not static; they are “continuously constructed and reconstructed as individuals engage in... Interaction with incumbents of alter roles” [11].

Like routines, roles can be conceptualized as patterns of action, but the logic of their enactment is different. Within a routine, the logic of enactment is analogous to control flow in a business process, where one action triggers the next [12]. Sequential triggering of actions within a routine gives rise to a recognizable pattern. To the extent that patient treatment is routinized, it should have a recognizable pattern of actions.

Within a role, however, the logic of what to do next may have little to do with the flow of the routine. For example, the office assistant who checks patient into the clinic serves each patient as they arrive. They perform roughly the same actions for each patient who arrives and they are not concerned with (or aware of) the rest of the process. The clinical technician who brings the patient to the room and takes vital signs has a similarly limited role. These specialized roles perform repetitive, recognizable patterns of actions, but they are driven by the functional requirements of the role. They see every patient, but they see only one part of the overall clinical routine.

Using data collected through participant observation, Kremser and Blagoev [1] analyze the patterns of action in an organization where consultants with specialized roles work on several concurrent projects. Kremser and Blagoev [1] note that the needs of each client workflow can change unexpectedly, as can the availability of the consultants. Changes in one workflow can cascade through the other workflows. Such cascading

interactions are common in outpatient clinics, where an unexpectedly difficult patient can tie up the clinical staff and delay the treatment of other patients.

We can contrast the complex service organization studied by Kremser and Blagoev [1] with other, less complex organizational forms, as in Fig. 1.

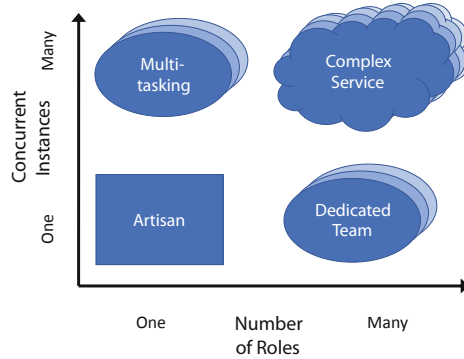


Fig. 1. Four kinds of role-routine ecology

The horizontal axis of Fig. 1 refers to the number of distinct roles involved in providing the service. For example, in a typical clinical visit, there may be a nurse, a physician and an office administrator, plus other roles. The vertical axis refers to the number of concurrent process instances (e.g., patients in the clinic at the same time). Using these two dimensions, four archetypal role-routine ecologies are easily identified.

- Artisan: One physician treats one patient at a time (e.g., sole practitioner).
- Dedicated team: a group of specialists treats one patient at a time (e.g., surgery).
- Multi-tasking: one nurse cares for several patients at the same time (e.g., in an inpatient hospital ward).
- Complex service: a team of specialists treats several patients at the same time (e.g., a typical outpatient clinic).

A wide variety of work processes fall into the category of complex service organizations, such as restaurants [13], professional service firms [1], software development teams [14], and the example we use here, outpatient medical clinics.

2.2 Concurrency in Complex Service Organizations

Concurrency is a pervasive aspect of the complex service organization. Concurrency can be formally defined in terms of Petri nets [15] which are widely used to represent business processes. In a Petri net, two events are *concurrent* if they occupy parallel paths in the network. When two or more events are concurrent, the specific sequence of their execution is irrelevant to the outcome of the process, as long as they are all completed. For example, van der Aalst [2] uses a Petri net to model a medical process where “lab

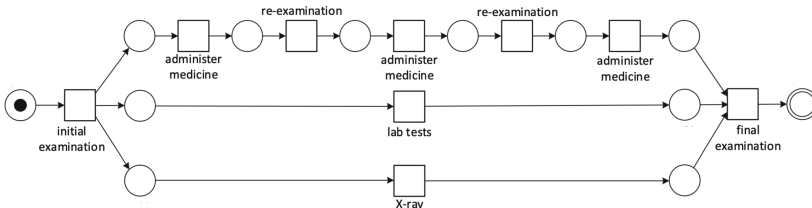


Fig. 2. Concurrency in a Petri net

tests” and “x-ray” occur concurrently with each other and with a series of other activities, as shown in Fig. 2.

In this paper, concurrency refers to actual overlap in time. As we use the term here, two events are concurrent if one starts before the other finishes. For example, in an outpatient clinic, two patient visits are concurrent if one patient arrives before the other patient leaves. From the perspective of the whole days’ work at the clinic, those patients could be seen in any order. However, the additional restriction of temporal overlap reflects the reality of clinical work. Treating multiple patients at the same time adds complexity to the clinical process because they are competing for resources [16].

2.3 Concurrency Within Process Instances

Traditional research on process management and process mining has emphasized the importance of concurrency within process instances [2]. While dependencies between multiple process instances is a recognized issue, mainstream theory and practice in process management generally treat process instances as independent. Process mining algorithms generally attempt to identify control flow within a process instance, rather than dependencies between process instances [3, 17].

2.4 Concurrency Between Process Instances

At the same time, research on process management has long recognized that workflows may be interdependent [18, 19]. When multiple instances of a process occur at the same time, they may compete for the same resources. Multiple instance patterns are defined to “describe situations where there are multiple threads of execution active in a process model which relate to the same activity” [20]. Common activities may also involve sharing common resources, such as a printer [16]. One approach to this general problem has been to treat multiple, concurrent process instances as a single, complex workflow. However, as Heinlein [19] notes, merging workflows (or messaging between workflows) faces a combinatorial explosion. If providers are serving n concurrent workflows, then “ 2^n variations of each workflow would be necessary in principle to describe its behaviour in every possible combination with n other workflows”.

The challenge of handling multiple process instances continues to be an active area of research in process management. Traganos, Spijkers, Grefen and Vanderfeesten [21] note that BPMN (Business Process Management Notation) lacks strong support for operations such as buffering, bundling, and unbundling physical objects in a manufacturing

workflow (such as parts on a palette). These types of operations are needed to handle multiple, concurrent process instances. Senderovich, Leemans, Harel, Gal, Mandelbaum and van der Aalst [22] analyze the use of event logs to discover queues. Suriadi, Wynn, Xu, van der Aalst and ter Hofstede [23] propose a method for discovering prioritization from event logs. Fahland, Denisov and van der Aalst [5] note that queueing for shared resources can introduce unexpected behavior in a process which is “particularly important for distributed systems with shared resources, e.g., one case can block another case competing for the same machine, leading to inter-case dependencies in performance.” Klijn, Mannhardt and Fahland [4] have proposed a graph-based framework for analyzing the inter-case dependencies involving actions and actors in digital trace data.

2.5 Competing Effects of Concurrency

Within a process instance, concurrency generally facilitates efficiency. When activities can be performed in parallel, it tends to increase the capacity of a work system. And when those activities can be performed in any sequence, it adds flexibility to a work system.

However, when there is concurrency between process instances, the effects are not so clear cut. The effects depend on the level of available resources and structure of the work system. If resources are limited, concurrent instances will be in competition [16]. The nature of that competition will be defined, in part, by the structure of the role-routine ecology in the organization. In an organization of artisans, each of whom works independently on a single process instance, resource constraints should be manifest in queuing [22]. In contrast, in a complex service organization, where multiple roles serve multiple clients at the same time, the effects of concurrency will depend on how work is coordinated, as well as resource constraints.

3 Research Context

To investigate this phenomenon, we analyze data from dermatology clinics at an academic medical center in the Northeastern U.S. We chose this setting because it provides a clear example of a complex service organization with multiple roles and multiple concurrent process instances (patient visits).

3.1 EHR Audit Trail Data

The EHR audit trail is an ideal resource for analyzing the clinical documentation process because every action by each provider who touches the clinical record is time-stamped and recorded. Providers include nurses, physicians, technicians, office assistants, insurance specialists, administrators, and others. The audit trail is not the full patient medical record; it is a separate database of who did what. It does not contain notes, test results, medications, billing information, costs or any other information about the content or outcomes of the medical services performed. For this study, we use audit trail data from the EPIC EHR system. The data traces the clinical documentation process for patient visits from three dermatology clinics from January 2016 through December 2017 for a total of 21,785 patient visits.

3.2 Concurrency in the Clinic

Concurrency is built into the physical layout of the clinics. In each of the dermatology clinics we studied, there were multiple exam rooms. Providers move between rooms, from patient to patient, as they do their work. For example, after a technician records pulse and blood pressure for one patient, they leave the room to perform some other tasks and someone else continues the visit with that patient. The overall workflow depends on which patient happens to be in the next room and what needs to be done. And of course, whatever gets done needs to be documented. In this way, the EHR documentation work is woven into the fabric of the medical work. The audit trail data provides a detailed record that we can use to examine the temporal structure of this fabric. It allows us to see two layers of concurrency in the clinic:

- Between patient visits, there is a great deal of concurrency. We can see this very accurately in the EHR audit trail data.
- Within patient visits, there is also some concurrency. We measure this with the EHR audit trail data, using the method described by Iqbal and Riek [24] but the measure is not perfect because many actions are not directly recorded.

When we view the event log for a single patient visit, it is easy to overlook the fact that there is more than one patient in the clinic at the same time. Idealized models of a process often assume that concurrent instances are independent. That is clearly not the case in medical practice. There are almost always multiple patients competing for time and attention. In our data, the average number of concurrent patients was 6.35. The maximum was 27.

4 Methods

We use the audit trail data to show how process duration is influenced by concurrency within and between process instances. To do so, we control for everything that might influence duration so we can more accurately estimate the effects of concurrency. To be clear, the data we analyze here was collected as part of a larger study, so we are relying on available metrics for this analysis.

4.1 Descriptive Statistics

Table 1 describes all the variables that we used in this analysis. We use a natural log transformation for all variables except the dummy variables. Table 2 provides the correlation for the main variables in the analysis.

Table 1. Descriptive statistics of variables.

Variables	Mean	sd	min	max	Description
<i>Duration</i>	8.32	0.53	0	10.15	Time between the first and last activity
<i>Concurrency WITHIN</i>	0.45	0.20	0	1	Level of concurrency among actors in the visit
<i>Concurrency BETWEEN</i>	1.88	0.51	0	3.34	Number of other visits that overlap in time with this visit
<i>NProviders</i>	1.61	0.20	1.38	2.40	Number of providers in visit
<i>NProcedures</i>	0.79	0.42	0	4.49	Number of procedures performed in visit
<i>Newbies</i>	0.07	0.26	0	1	Any new employees on this visit? (0/1)
<i>FirstVisit</i>	0.55	0.50	0	1	Is this the first visit for this patient? (0/1)
<i>Diagnosis</i>					Complexity of diagnosis (3 levels)
<i>CPT Code</i>					Billing code for level of service (5 levels)
<i>Clinic</i>					Which clinic? (3 levels)
<i>Month</i>					Which month? (12 levels)

Table 2. Spearman correlation matrix for variables (n = 21,785)

	(1)	(2)	(3)	(4)	(5)	(6)	(7)
1. Duration	1						
2. <i>WITHIN</i>	-0.522	1					
3. <i>BETWEEN</i>	0.408	-0.399	1				
4. NProcedures	0.082	-0.024	0.055	1			
5. NProviders	0.261	-0.185	0.094	0.063	1		
6. Newbies	0.025	-0.011	-0.036	0.002	0.023	1	
7. FirstVisit	-0.014	0.017	-0.018	0.062	-0.022	0.020	1

5 Model Specification and Findings

To examine the effect of concurrency on duration of the patient visits, we used the main empirical specification as follows:

$$\begin{aligned} \text{Duration} = & \beta_0 + \beta_1(\text{WITHIN}) + \beta_2(\text{BETWEEN}) + \beta_3(\text{NProcedures}) \\ & + \beta_4(\text{NProvider}) + \beta_5(\text{Newbies}) + \beta_6(\text{First}_{\text{visit}}) + \alpha + \lambda + \gamma + \delta \end{aligned}$$

where *Duration* is the log of duration of the patient visit, concurrency *WITHIN* is the level of overlap between actions within each visit and concurrency *BETWEEN* is the number of other patients in the clinic. We control for a number of other variables, including the number of providers (*NProviders*), the number of procedures performed during the visit (*NProcedures*), the involvement of new workers (*Newbies*), and whether this was the first visit for this patient (*First_Visit*). We further add fixed effects in the model to account for heterogeneity due to the clinic (α), diagnosis complexity (λ), level of service (γ), and monthly seasonality (δ).

Table 3 reports the estimated effects of concurrency on duration in the patient visits. In column (1), we use only the control variables and fixed effects. In columns (2) and (3), we show to add the stepwise effects of concurrency within and between visits. In column (4), we show the full model. These are standardized coefficients so we can compare their relative magnitudes.

As expected, each aspect of concurrency is associated with a significant change in the duration of patient visits. Concurrency within visits speeds up the work, while concurrency between slows it down. Concurrency within visits is the largest effect, roughly three times the size of concurrency between visits. However, when we take both effects into account at the same time, as in column (4), their magnitudes are somewhat reduced.

The control variables also provide some interesting insights. For example, the number of providers involved in a visit increases the duration as much as concurrency between visits. This is because each type of provider has a specialized role. In the simplest (fastest) visits, the patient interacts with 1–2 clinical staff members. If a visit requires the attention of more staff members, it is likely to involve a more elaborate and time-consuming process. Interestingly, the number of procedures performed (e.g., freezing a wart) has a comparatively small effect on duration. Likewise, the involvement of individuals who are new to their jobs (newbies) has a small positive effect. However, contrary to our expectations, first-time visits are not longer than follow-up visits.

Table 3. OLS regression results (standardized coefficients)

	(1)	(2)	(3)	(4)
Variables	Controls	Within	Between	Both
WITHIN		-1.5323*** (0.0492)		-1.1912*** (0.0431)
BETWEEN			0.4394*** (0.0096)	0.3908*** (0.0089)
NProviders	0.5111*** (0.0226)	0.4666*** (0.0220)	0.4476*** (0.0217)	0.4200*** (0.0213)
NProcedures	0.0581*** (0.0081)	0.0640*** (0.0078)	0.0649*** (0.0077)	0.0688*** (0.0074)
Newbies	0.0550*** (0.0128)	0.0505*** (0.0124)	0.0722*** (0.0121)	0.0668*** (0.0118)
FirstVisit	-0.0085 (0.0092)	0.0026 (0.0087)	-0.0072 (0.0084)	0.0014 (0.0081)
Constant	7.1051*** (0.1142)	7.4038*** (0.1064)	6.5918*** (0.1043)	6.8808*** (0.0985)
Observations	21,800	21,800	21,800	21,800
R-squared	0.1400	0.2026	0.2501	0.2865
LOS fixed effects	YES	YES	YES	YES
Clinic fixed effects	YES	YES	YES	YES
YM dummies	YES	YES	YES	YES
Diagnosis effects	YES	YES	YES	YES

Robust standard errors in parentheses

*** $p < 0.001$, ** $p < 0.01$, * $p < 0.05$

6 Discussion

Without question, concurrency is crucial to the true fabric of organization, but concurrency occurs in layers, within and between processes. In terms of the fabric metaphor, organizational fabric has multiple layers and they are loosely stitched together. In simple organizations, where processes and process instances are independent, it is relatively easy to understand the effects of concurrency within a process. But in complex service organizations, where multiple providers serve multiple concurrent clients, it is not so easy, because concurrency within each process instance interacts with concurrency between process instances. We can see in Table 2 that there is a strong negative correlation ($r = -0.399$) of concurrency within and between. When there are more patients in the clinic, the clinical staff are spread thin. They are less likely to be working on the same patient at the same time. They are more likely to be working on different patients.

This relationship will be different in other kinds of organizations, but it points to the possibility of multi-layer interactions.

6.1 What Controls the Flow?

An important insight from the role-routine ecology is that process flow can be controlled by multiple, competing priorities or logics [25, 26]. The next action or event is not necessarily triggered by actions or events in the same case (i.e., the same patient visit). Rather, it could be triggered by a pattern of action implied by the roles of the clinical staff.

Routines can be understood in terms of a sequential, control flow logic, where one event triggers the next. For some roles, “control flow” works very differently. For the office staff who check patients in, the work consists of checking in the next patient. For the clinical technician, the work consists of getting vital signs for the next patient. These specialized roles contribute the same steps to each patient visit. These roles just take the next patient in the queue. Similarly, as the nurses and physicians work their way around the exam rooms, from one patient to the next, they are executing role-based patterns of action. The fact that patients can be seen in any order (the broad definition of concurrency) adds flexibility to the workflow, but it also adds complexity to the event log.

6.2 Emergent Complexity and Model Quality

In research on organizational routines, there has been growing interest in the antecedents and consequences of complexity [27–29]. This research treats process complexity as the emergent product of situated actions. In research on process mining, Augusto, Mendling, Vidgof and Wurm [30] demonstrate that event log complexity can influence the quality of models discovered through conventional process mining. Their analysis starts from the complexity event log. Here, we are stepping back to consider why some event logs are more complex than others.

In a simple role-routine ecology, role logic and routine logic are likely to be aligned. For example, in a simple organization of artisans, where one individual performs one process instance at a time, the event log for each process instance will be independent of other instances. In this idealized case, the pattern of action for the role and the routine should be the same. This is the best-case scenario for process mining and the discovery of concurrency within the process. However, as we add multiple roles and concurrent process instances, the best-case scenario breaks down. In a more complex role-routine ecology, there is inevitably some conflict between roles and routines. To the extent that the event log is an emergent product of these competing logics, playing out over concurrent process instances, it will be more difficult to model.

7 Conclusion

As process mining capabilities become more mature, there is growing interest in more sophisticated applications of process discovery, such as digital twins of organizations

[7, 8]. Such models seem plausible for the best-case scenario when concurrency is limited to within process instances, and the control flow is governed by a single, uniform logic. However, in more complex organizations, where there are interdependent process instances (not to mention interdependent processes) and competing logics for each of them, discovering and modeling the true fabric of organization is inherently more difficult.

Acknowledgements. This material is based upon work supported by the National Science Foundation under Grants No. SES-1734237 and BCS-2120530. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. This research was also supported in part by University of Rochester CTSA (UL1 TR002001) from the National Center for Advancing Translational Sciences (NCATS) of the National Institutes of Health (NIH). The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. We are grateful to the reviewers for their comments.

References

1. Kremser, W., Blagoev, B.: The dynamics of prioritizing: how actors temporally pattern complex role-routine ecologies. *Adm. Sci. Q.* **66**, 339–379 (2021)
2. van der Aalst, W.M.: Concurrency and objects matter! Disentangling the fabric of real operational processes to create digital twins. In: Cerone, A., Ölveczky, P.C. (eds.) *ICTAC 2021*. LNCS, vol. 12819, pp. 3–17. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-85315-1_1
3. van der Aalst, W.M.: Business process management as the “killer app” for Petri nets. *Softw. Syst. Model.* **14**, 685–691 (2015)
4. Klijn, E.L., Mannhardt, F., Fahland, D.: Classifying and detecting task executions and routines in processes using event graphs. In: Polyvyanyy, A., Wynn, M.T., Van Looy, A., Reichert, M. (eds.) *BPM 2021*. LNBP, vol. 427, pp. 212–229. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-85440-9_13
5. Fahland, D., Denisov, V., van der Aalst, W.: Inferring unobserved events in systems with shared resources and queues. *Fund. Inform.* **183**, 203–242 (2021)
6. Hodgson, G.M.: Institutions and individuals: interaction and evolution. *Organ. Stud.* **28**, 95–116 (2007)
7. Park, G., Van Der Aalst, W.M.: Realizing a digital twin of an organization using action-oriented process mining. In: *2021 3rd International Conference on Process Mining (ICPM)*, pp. 104–111. IEEE (2021)
8. van der Aalst, W.M., Hinz, O., Weinhardt, C.: Resilient digital twins. *Bus. Inf. Syst. Eng.* **63**, 615–619 (2021)
9. Feldman, M.S., Pentland, B.T.: Reconceptualizing organizational routines as a source of flexibility and change. *Adm. Sci. Q.* **48**, 94–118 (2003)
10. Rerup, C., Feldman, M.S.: Routines as a source of change in organizational schemata: the role of trial-and-error learning. *Acad. Manag. J.* **54**, 577–610 (2011)
11. Turner, J.H.: *Handbook of Sociological Theory*. Handbooks of Sociology and Social Research, p. 730. Springer, Dordrecht (2006)
12. Cohen, M.D., Bacdayan, P.: Organizational routines are stored as procedural memory: Evidence from a laboratory study. *Organ. Sci.* **5**, 554–568 (1994)

13. Tanizaki, T., Shimmura, T.: Modeling and analysis method of restaurant service process. *Proc. CIRP* **62**, 84–89 (2017)
14. Dikert, K., Paasivaara, M., Lassenius, C.: Challenges and success factors for large-scale agile transformations: a systematic literature review. *J. Syst. Softw.* **119**, 87–108 (2016)
15. Murata, T.: Petri nets: Properties, analysis and applications. *Proc. IEEE* **77**, 541–580 (1989)
16. Mangler, J., Rinderle-Ma, S.: Rule-based synchronization of process activities. In: 2011 IEEE 13th Conference on Commerce and Enterprise Computing, pp. 121–128. IEEE (2011)
17. Rojas, E., Munoz-Gama, J., Sepúlveda, M., Capurro, D.: Process mining in healthcare: a literature review. *J. Biomed. Inform.* **61**, 224–236 (2016)
18. Alonso, G., Agrawal, D., El Abbadi, A.: Process synchronization in workflow management systems. In: *Proceedings of SPDP 1996: 8th IEEE Symposium on Parallel and Distributed Processing*, pp. 581–588. IEEE (1996)
19. Heinlein, C.: Workflow and process synchronization with interaction expressions and graphs. In: *Proceedings 17th International Conference on Data Engineering*, pp. 243–252. IEEE (2001)
20. van der Aalst, W.M., Ter Hofstede, A.H., Kiepuszewski, B., Barros, A.P.: Workflow patterns. *Distrib. Parallel Databases* **14**, 5–51 (2003)
21. Traganos, K., Spijkers, D., Grefen, P., Vanderfeesten, I.: Dynamic process synchronization using bpmn 2.0 to support buffering and (un) bundling in manufacturing. In: Fahland, D., Ghidini, C., Becker, J., Dumas, M. (eds.) *BPM 2020. LNBP*, vol. 392, pp. 18–34. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58638-6_2
22. Senderovich, A., Leemans, S.J.J., Harel, S., Gal, A., Mandelbaum, A., van der Aalst, W.M.P.: Discovering queues from event logs with varying levels of information. In: Reichert, M., Reijers, H.A. (eds.) *BPM 2015. LNBP*, vol. 256, pp. 154–166. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-42887-1_13
23. Suriadi, S., Wynn, M.T., Xu, J., van der Aalst, W.M., ter Hofstede, A.H.: Discovering work prioritisation patterns from event logs. *Decis. Support Syst.* **100**, 77–92 (2017)
24. Iqbal, T., Riek, L.D.: A method for automatic detection of psychomotor entrainment. *IEEE Trans. Affect. Comput.* **7**(1), 3–16 (2015)
25. Thornton, P.H., Ocasio, W.: Institutional logics. *Sage Handb. Organ. Inst.* **840**, 99–128 (2008)
26. Reay, T., Hinings, C.R.: Managing the rivalry of competing institutional logics. *Organ. Stud.* **30**, 629–652 (2009)
27. Goh, K.T., Pentland, B.T.: From actions to paths to patterning: toward a dynamic theory of patterning in routines. *Acad. Manag. J.* **62**, 1901–1929 (2019)
28. Hansson, M., Hærem, T., Pentland, B.T.: The effect of repertoire, routinization and enacted complexity: explaining task performance through patterns of action. *Organ. Stud.* 01708406211069438 (2021)
29. Danner-Schröder, A., Ostermann, S.M.: Towards a processual understanding of task complexity: constructing task complexity in practice. *Organ. Stud.* **43**, 437–463 (2022)
30. Augusto, A., Mendling, J., Vidgof, M., Wurm, B.: The connection between process complexity of event sequences and models discovered by process mining. *Inf. Sci.* **598**, 196–215 (2022)