Towards Modelling and Simulation of Organisational Routines

Phub Namgay and David Johnson

Department of Informatics and Media, Uppsala University, Box 513,
751 20 Uppsala, Sweden
{phub.namgay,david.johnson}@im.uu.se

Abstract. Organisational routines are repetitive, recognisable patterns of interdependent action by human and digital actors to accomplish tasks. Routine Dynamics is a theoretical base that informs discussion and analysis of such routines. We note that there is a knowledge gap in the literature on organisational routines to consolidate the constructs and ontologies of routines into an abstract data model. In this paper, we design and implement a data model of routines using the Unified Modelling Language. We present a demonstration to illustrate our data model's use, and how one can then use instantiations of the model to analyse and simulate organisational routines based on real-world data. This example examines recurrent patterns of action inferred from data in the GitHub issue tracking system about the open-source software project, *scikit-learn*. Our study extends the theoretical/empirical understanding and knowledge base of Routine Dynamics by laying the groundwork towards examining organisational routines from a model-driven perspective that gives rise to simulating the dynamics of routines.

Keywords: organisational routines · Routine Dynamics · modelling and simulation.

1 Introduction

Organisational routines are defined as '... repetitive, recognizable patterns of interdependent actions, carried out by multiple actors.' [1]. Routine Dynamics [1,2] is an increasingly applied theoretical perspective of organisational routines that account for recurrent patterns of interdependent action by sociotechnical actors in digital ecosystems [3]. It is a foundational theory for studying routines in workflows and work practices, particularly in the fields of organisational science and information systems research. Numerous studies on routines thus far, such as the effect of emergent routines in open-source software (OSS) development [4], have fostered the development of Routine Dynamics as a robust tool to investigate processes in sociotechnical phenomena. However, there has yet to be a consolidation and formalisation of Routine Dynamics' constructs and ontologies into a standard model of routines, and thus there lie uncertainties when dealing with related data. The application of Routine Dynamics to research

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63783-4_27

and practice is typically done in an ad-hoc fashion, where different individual studies re-interpret and apply the theoretical concepts heterogeneously. As a consequence, comparisons of studies relating to organisational routines becomes cumbersome, and there are no agreed upon approaches to studying routines and related data.

The paper is structured as follows: In Section 2, we discuss background literature relating to organisational routines, and describe our motivations for the work reported in this paper. In Section 3, we provide an overview of our data model of Routine Dynamics. In Section 4, we demonstrate how we can instantiate the data model using a popular OSS project as the subject of study to perform analysis and simulations of routines. In Section 5, we discuss the opportunities and challenges of modelling routines, and finally conclude the paper in Section 6.

2 Background and Motivation

Routines are essential to organised work [1]. They are observed behaviours following recognised action patterns through multiple interdependent actors [30]. The theory underscores action over actors and does not distinguish between human and material agency. Feldman and Pentland [1] note, 'A pattern of action that occurs only once is not a routine.' Sociotechnical actors acquire knowledge [29] in enacting routines to accomplish tasks through learning from experience. Thus far, the literature on routines has primarily focused on theoretical and empirical studies. Hayes, Lee and Dourish [5] explored the enactment of processes and routines via a narrative network perspective [6] with narrative fragments as nodes bridged by an action [7]. The processual dynamics of routines have been studied to account for digitised processes through a combination of humans and material agents [8]. Studies on routines investigating an invoice processing system as 'patterns of action' suggest that the action patterns differ across organisations despite using the same systems [9]. Gaskin et al. [10] conducted a study examining and modelling the lexicons of sociomaterial routines using the sequence analysis. Table 1 summarises the key concepts from Routine Dynamics theory that capture routines by enabling or constraining the enactment of processes [8].

In our review of the literature, we do not find any evidence of research that has explored routines from the perspective of standardising the approaches to expressing data about routines for the purposes of modelling and simulation. Therefore, we set out to build a first model of Routine Dynamics.

3 Proposed Data Model for Routine Dynamics

Our proposed data model of Routine Dynamics draws on the concepts extracted from the routines literature summarised in Table 1, where we use an object-oriented design formalisation expressed in UML. Figure 1 shows our data model

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63783-4_27

Table 1. Concepts and definitions of organisational routines.

Concept	Definition
Routines	Recurrent, recognisable patterns of action enacted by multiple interdependent actors [1], such as routines to resolve issues on the GitHub issue tracking system
Actor	Human or digital agents that perform a task, which implies the agency of agents [1], such as different sociotechnical actors involved in the GitHub issue tracking system [11]
Action	Steps taken to accomplish a task and things actors do, such as different phases and iterations for resolving issues on GitHub
Action pattern	Steps to accomplish a task that repeats over time, such as repetitive action patterns to flag, report, and resolve issues on the GitHub issue tracking system
Activity	Basic unit of carrying out a single function by actors and produces an outcome in routines [10], and activities in routines vary in structure and dynamics
Ostensive	Abstract codified repetitive, recognisable patterns of action [9], which embody the structure of routines [1,4] and are implicit [7], such as codified standard operating procedure, manual or taken-for-granted norms that underpin actual performances of routine work (see [5])
Performative	Actual performance of routines by interdependent actors, which bring routines to life across spatiotemporal dimensions [1], such as human actors resolving issues on the GitHub issue tracking system
Generativity	Create or produce something new or adapt existing routines through the agency of sociotechnical actors in routines, and thus 'generative capacity' [7,12,13]
Intermediary	An actor involved in maintaining connections or relaying data or information with minimal performative power while being engaged in a task [13], such as an application programming interface (API) that is passively involved in facilitating connection and communication on GitHub
Mediator	An actor that transforms, translates, and modifies meaning in action [13] in a network of processes to accomplish tasks, such as a contributor on GitHub who is actively involved in analysing and resolving issues
Live routine	Repetitive, recognisable patterns of action enacted by humans [5,14] through an individual agency to improvise and learn from experience, such as manual processes to resolve issues on GitHub issue tracking system
Dead routine	Recurrent patterns of action emerging from artefacts that are characterised as 'rigid, mundane, mindless, and codified' [15], such as action patterns of algorithmic agents for resolving issues through event logs of user communities and contributors on GitHub issue tracking system

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63783-4_27

as a UML class diagram based on the concepts from Table 1 and how the concepts are interrelated.

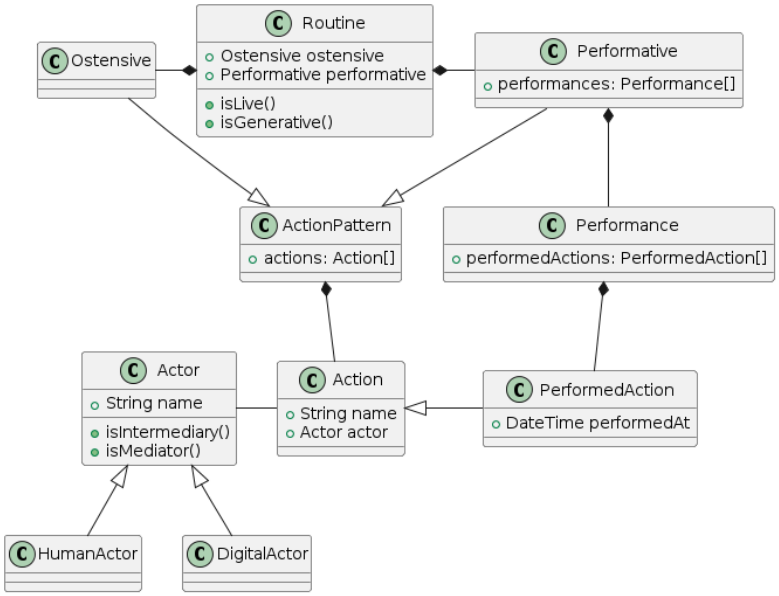

Fig. 1. UML class diagram of the routines data model, where concepts are drawn from literature as described in Table 1.

Routines embody a duality of structure and agency. An ostensive aspect embodies codified structure [5]. The `Routine` class is composed of both `Ostensive` and `Performative` classes. Routines can be guided by one or many ostensive facets and performative facets. For example, standard operating procedures, norms, and rules are ostensive features of routines as they evoke a simplified view of tasks in practice. The actual performance of routines by interdependent sociotechnical actors represents agency [1]. Ostensive and performative share a mutual recursive association [1,7], where the action patterns that compose of each influence each other in practice. Consequently, we model the `Ostensive` and `Performative` as extensions of an `ActionPattern`, where `ActionPatterns` are a composition of a sequence of `Actions`. `Actions` are carried out by `Actors`, which may be `HumanActors` (representing real people performing an action) or `DigitalActors` (representing a non-human agents performing an action, often a computer program). The `Performative` aspect concerns actual `Performances`, which are composed of a sequence of `PerformedActions`. One can distinguish an `Action` and `ActionPattern` from a `PerformedAction` and `Performance` by thinking of the former as prototypical patterns, while the latter are indicative of actual instances of a routine having been in action. Hence, `PerformedActions`

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63783-4_27

have a timestamp while `Action`s do not. Performativity closely connects with generativity. Human and digital actors have an endogenous capacity to generate innovative outcomes, retain novel action patterns, or adapt action patterns [4,13]. The `Routine` class therefore contains the `isGenerative()` interface to enable implementations of generativity assertion. However, it is not a compulsion for routines to manifest generativity.

In routines, an actor can be a human or digital artefact, as indicated by the 'is a' inheritance relationship. The `Actor` class holds some common properties shared by its child classes. A task is taken up by actors, and subsequent routines to perform the task can be 'live' routines or 'dead' routines [14,15] or an interplay of the two. The `Routine` class therefore contains an interface to enable the implementation of an `isLive()` method to allow implementations of assertions on the type of routine (live or dead). For example, GitHub collaborators can enact 'live' routines for resolving issues on repositories, projects, or codes, where each and every `Action` is carried out by a `HumanActor`. In contrast, the GitHub platform's software actively facilitates services via autonomous bots, such as continuous integration, where a `Routine` is performed entirely by `DigitalActor`. Depending on their generative potential, an actor can act as an intermediary or mediator in task routines [9,13], and thus `Actor`s also contain corresponding assertion interfaces in `isIntermediary()` and `isMediator()`. In itself, digital artefacts can act as a mediator. However, when its service is used in another system, it might serve as a passive digital agent, thus an intermediary role in routines [13]. How these interfaces are implemented is intentionally left undefined in the data model in order to allow flexibility of interpretation on the concepts that are harder to describe as data tangibly.

4 Demonstration of Routines Data Model In-use

In order to demonstrate our routines data model, we replicated the approach used by Deng et al. [4] for analysing routines inferred from issues posted in the GitHub issue tracking system. In particular, we use GitHub issues as trace data [16] that record specific action patterns, and thus we can discover recurrent action patterns (i.e. performative routines) that are used to resolve issues in projects on GitHub. When able to identify performative routines in trace data, we can then utilise extant quantifiable data relating to the performances for further analysis or modelling. In this section, we present how we map GitHub's representation of issues and associated events into the routines data model and an example of how one can then use the instantiated data model for simulation purposes.

4.1 *Scikit-learn* development as a data source

Scikit-learn is a robust Python library for machine learning and statistical modelling [11]. As an OSS project hosted on GitHub, *scikit-learn* has an active development community with code contributions going back as far as 2010 and over

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63783-4_27

33,000 issues and pull requests tracked. The issue resolution process on GitHub [17] can be considered as a transparent and accountable assemblage of humans, integrated development environments, algorithmic processes, APIs, data, norms, and policies in a constant interplay as 'live' and 'dead' routines [14]. Resolving issues through discussion ensures that all the activities are legitimate from authorised users, thereby ensuring technical integrity in open environments [18]. The issue resolution routines on the *scikit-learn* repository present an ideal real-world case to map the dynamics of sociotechnical actors' patterns of action for undertaking collaborative tasks to our routines data model. In addition, such a study also presents an opportunity to examine the ostensive, performativity, and generativity of human and digital actors in practice.

Data related to the routines for resolving issues in an OSS project on GitHub can be viewed from two perspectives through the lens of Routine Dynamics theory. Ostensive routines can be thought of as the idealised normative examples of recurrent action patterns, often determined through the qualitative analysis of authoritative sources of data, such as participatory or direct observations, documentation and expert interviews. With respect to OSS contributions and issue tracking, guidelines provided by GitHub themselves or project-specific documentation, as illustrated in Table 2, could be qualitatively analysed to determine the ostensive aspect of issue resolution routines.

Table 2. Documents that provide qualitative data to infer the ostensive aspect of *scikit-learn* repository routines (see [1,5]).

Policy document	Description
Scikit-learn governance and decision-making [19]	The document formalises the governance mechanism for managing the *scikit-learn* project and clarifies decision-making and interaction among various elements in the *scikit-learn* community. The document establishes a decision-making structure that considers feedback from all community members.
Scikit-learn community code of conduct [20]	The document adopts a guideline to define community standards to foster a welcoming and inclusive *scikit-learn* project. It also facilitates a healthy and constructive community and social atmosphere for projects to develop and achieve their desired goals.
GitHub Docs - Issues [21]	Prescriptive guidelines for reporting issues for ideas, feedback, tasks, or bug reporting on GitHub.
GitHub Docs - Collaborating with pull requests [22]	Prescriptive guidelines for proposing and reviewing changes in pull requests on GitHub.

On the other hand, the performative aspect of routines looks at how routines are actually enacted or performed, the patterns of which are ascertained by observing so-called performances. Ostensive routines theoretically guide how a routine should be performed, but this does not mean the corresponding performances conform to the norms in practice. These two aspects together (i.e. the

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63783-4_27

ostensive and the performative) allow us to analyse routines and their dynamics deeply. Routine performances may be evidenced through, again, participatory or direct observation, logging (i.e. diaries/journals or digital logs), and increasingly by computationally processing digital trace data. In the context of this exemplar, as in [4], we use GitHub issues themselves as the trace data from which to infer routine performances.

4.2 Research replication using our data model

Our demonstration analysis is inspired by the work reported in [4], where Deng et al. show the impact of routine change in OSS development. They use a methodological approach that leverages digital trace data from GitHub to focus on how changes in emergent development routines influence OSS project popularity. The authors selected a stratified sample of 271 OSS projects based on their popularity, measured via forks and stars, to ensure a diverse representation of projects, ultimately building a dataset of over 20 million events and inferring over 3 million performances. They then use sequence analysis [23], a technique similar to gene sequence analysis in bioinformatics, but where categorical data are organised chronologically, and hidden Markov models (HMM) [24] to identify and quantify emergent routines and their changes within these projects. Routine diversity is then measured by the variety of routines, while routine change is assessed both in terms of its magnitude and frequency. What their study demonstrates is that large-scale computational analysis of trace data can be used to generate knowledge about routines. The authors use the GitHub API's data model of events as the units of observation by which to construct routine performances and thus infer 'contextualised routines' that describe specific patterns of action, where they then quantitatively analyse these patterns.

In Deng's paper, the authors present an example of 'contextualised routines' for a single OSS project, *Angular.js*. To demonstrate the application of our data model, we replicate this analysis presented in that paper but for the routines found by mining *scikit-learn*'s issues.

4.3 Modelling routines in the *scikit-learn* OSS project

We use the GitHub REST API to extract issues relating to software development from the *scikit-learn* OSS project repository. This was done using the PyGitHub v2.2.0 library [25], which provides access to the REST API via typed interactions in Python programming code. We collected 25,799 GitHub issues and pull requests marked as closed (as of 19 April 2024). The GitHub REST API provides access to download each issue, indicated with an `IssueEvent` object. Each `IssueEvent` has a related timeline of constituent events that is accessed using PyGitHub with a `.get_timeline()` method. This method gives us a chronological sequence of events, the types of which are listed in Table 3, and their metadata such as related users, timestamps, state, etc.

By iteratively retrieving issues and their timelines using the GitHub API, and retrieving their constituent event objects, we construct a dataset of 617,440

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63783-4_27

Table 3. GitHub event types relating to development teamwork, as per [4], and related GitHub data entities, and the corresponding mapping into our routines data model classes.

GitHub API model	Inferred action	Routines model
`IssueEvent`	Issue is created by a user (human or digital)	`PerformedAction`
`PushEvent`	Code is committed and pushed to a repository	`PerformedAction`
`CloseEvent`	Issue or pull requests is closed	`PerformedAction`
`ReopenEvent`	Issue or pull request that had been closed is reopened	`PerformedAction`
`PullRequestEvent`	User request new code to be pushed to a repository	`PerformedAction`
`CommentEvent`	Comment is created on issue or pull request	`PerformedAction`
`CommitCommentEvent`	Comment is created on a commit	`PerformedAction`
`PullRequestReviewEvent`	Review comment is created on a pull request	`PerformedAction`
`MergeEvent`	Pull request is accepted and code is merged to a repository	`PerformedAction`
`SubscribeEvent`	Issue or pull request is subscribed to by a user	`PerformedAction`
`UnsubscribeEvent`	Issue or pull request is unsubscribed by a user	`PerformedAction`
`ReferenceEvent`	Issue or pull request is referenced	`PerformedAction`
`MentionEvent`	User user is mentioned	`PerformedAction`
`AssignEvent`	Issue is assigned to a user	`PerformedAction`
`Timeline`	Collection of events ordered temporally	`Performance`
`User`	A GitHub user such as a human contributor	`HumanActor`
`User`	A GitHub user that is a bot	`DigitalActor`
	Recurrent action pattern derived from performances	`Performative`

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63783-4_27

events, spanning from 31 August 2010 to 17 April 2024, that we map into 25,799 `Performance`s made up of 617,440 `PerformedAction`s as expressed in our routines data model. From these 25,799 `Performance`s, we identify 8,474 recurrent `ActionPattern`s as performative routines (similar to [4]'s presented contextualised routines). It was done by identifying and reducing the repeated `PerformedAction` types to describe patterns of `Action` that occur in a particular chronological sequence. For each `PerformedAction`, we also identify and link an `Actor`, either human (`HumanActor`) or digital artefact (`DigitalActor`), as found in the GitHub event's metadata referencing a GitHub user. Table 4 summarises the top 10 most commonly identified performative routines from *scikit-learn*'s issue data.

Table 4. Summary statistics about the top 10 occurring recurrent action patterns (performative routines) from *scikit-learn* issues. **C = commented, CL = closed, O = opened, L = labelled, RV = reviewed, M = mentioned**.

Recurrent action pattern	No. events	No. comments	No. unique actors	Avg. duration
O/C+/CL	26007	11764	1735	105 days 12:33
O/CL	14000	2308	1166	24 days 10:57
O/C/CL	13439	3442	1113	55 days 04:41
O/L/C/CL	3636	676	391	78 days 17:36
O/RV+/CL	5084	189	260	4 days 07:45
O/RV/CL	3246	203	295	3 days 02:17
O/L+/RV/CL	3739	114	184	4 days 14:45
O/L+/RV+/CL	4544	72	149	5 days 13:57
O/C+/M/C+/CL	6462	3060	408	165 days 13:57
O/L/C+/CL	3203	1214	405	76 days 09:19

While the routines can be expressed as patterns in the style shown in Table 4, we can further look at the performative routines as stochastic models. Given their nature, the representation of routines as a sequence of actions means that we can straightforwardly model routines from the perspective of states and transitions, as simple Markov chains. Given that we have observations of routine performances represented in our data model, we can calculate transition probabilities from action-to-action and populate a transition matrix accordingly (see Table 5 and corresponding visualisation in Figure 2). Consequently, we can now perform simulations of routines by running the Markov chain.

Modelling routines as Markov chains provides further insights into how routines are performed based on real-world observations, where we can use the simulations to better understand the effects of different sequences of actions to fulfil a given routine. Taking this further, we may decide to model subsets of routines for comparative study, and can also integrate other data into the modelling and simulation, such as conformance to the ostensive routines (i.e. comparing distances between action patterns of the ostensive and performative), sequence variability, effects of different actor types and roles, performance factors and so

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63783-4_27

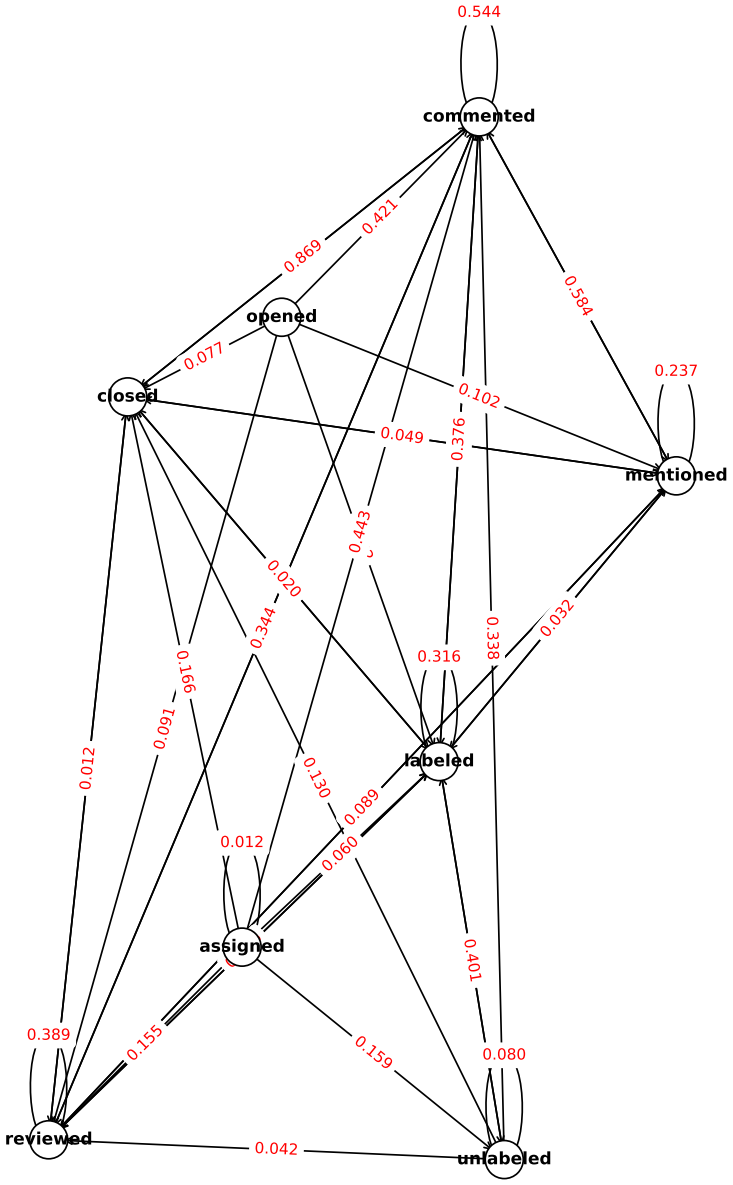

Fig. 2. Visualisation of the Markov chain model of issue resolution routines in *scikit-learn*, based on the transition probabilities shown in Table 4.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63783-4_27

Table 5. Transition matrix showing probabilities of the state transitions between different routine actions in *scikit-learn* development issues, based on the observed trace data from GitHub.

	commented	opened	labeled	reviewed	assigned	unlabeled	closed	mentioned
commented	0.5444	0.0000	0.0177	0.0505	0.0018	0.0066	0.0864	0.2926
opened	0.4210	0.0000	0.3018	0.0909	0.0019	0.0051	0.0770	0.1023
labeled	0.3755	0.0000	0.3159	0.2006	0.0024	0.0390	0.0563	0.0102
reviewed	0.3437	0.0000	0.0461	0.3887	0.0026	0.0069	0.1880	0.0240
assigned	0.4435	0.0000	0.0601	0.1555	0.0124	0.1590	0.1661	0.0035
unlabeled	0.3375	0.0000	0.4014	0.0422	0.0049	0.0804	0.1300	0.0036
closed	0.8691	0.0000	0.0195	0.0124	0.0000	0.0080	0.0089	0.0821
mentioned	0.5835	0.0000	0.0317	0.0894	0.0013	0.0082	0.0486	0.2374

on, or other underlying processes that are not directly observable perhaps using HMMs. Uncertainty modelling becomes vital when considering the performative aspect of routines.

5 Discussion

5.1 Opportunities for modelling routines

The uptake of Routines Dynamics in the mainstream benefited in expanding the constructs and ontologies of routines. However, using an established modelling framework to model routines has not received much scholarly attention. There remains much uncertainty in the interpretation of the constructs in Routine Dynamics [1,13] since they are abstract and complex, especially to novice researchers. As such, we argue that the formalised representation of routines as a data model may ease the comprehension of routines' ontologies, structure, and behaviour. Moreover, modelling enables one to have a deeper insight into the underlying intricate dynamics and assumptions of real-world routines. The normative approach of discourse on Routine Dynamics is found inadequate, especially adopting as a theoretical framework for researchers or analytical tools for practitioners. Therefore, model-driven approaches to studying routines aid in grasping the inner workings and multifaceted dynamics of routines, as illustrated in the analyses and modelling in Tables 4 and 5, and Figure 2. Indeed, such a conceptual representation and physical simulation allows one to visualise the duality of routines—structure and agency of sociotechnical actors. Due to its abstractness, some core constructs in Table 1 of routines are demanding to think of in agentic terms [1]. Thus, the UML [26,27] perspective to examine and model the structural and behavioural aspects of routines unfolding in the real world adds a novelty to the knowledge base of routines.

5.2 Challenges of modelling routines

The routines for accomplishing tasks will likely change in a dynamic environment with multi-agent systems [3]. Given the autonomous, decentralised, and

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63783-4_27

heterogeneous nature of actors in digital ecosystems, capturing routines enacted by humans or encoded in digital artefacts and corresponding micro-level knowledge embedded in patterned action is difficult. Winter [28] notes, 'knowledge advances cumulatively'. Hence, abstract properties of sociotechnical actors' routines such as expertise [1], knowledge [29], repertoire [30], and memory [4] in digitised processes are challenging to grasp and represent in a model. Additionally, one cannot take action patterns observed in a phenomenon at face value and treat them as routines without understanding context and questioning assumptions. The routines data model presented in this study is not static and can evolve as new constructs of routines are added to the knowledge base and refined over time.

6 Conclusion

This study unpacked routines' intricate structure and dynamics for resolving GitHub *scikit-learn* repository issues to develop a routines model using UML. Routines in an organisational environment entail complex agency, action, and interplay of interdependent sociotechnical actors. The UML class diagram captures routines' structural and behavioural facets. The knowledge base of routines lacks scholarly discussion and articulation of modelling routines that consolidate the constructs and ontologies into a conceptual, logical, and physical model. This study demonstrated that modelling frameworks such as UML could provide such an account of routines to further our understanding of and augment the knowledge base on routines. We suggest researchers and practitioners employ a modelling and simulation perspective to investigate the constructs, structure, and dynamics of routines. The source code and data for this paper is found in: https://github.com/UppsalaIM/Towards_Modelling_Organisational_Routines/.

Acknowledgements. Phub Namgay is supported in part by the Swedish Research School of Management and IT (Forskarskolan management och IT).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Feldman, M.S., Pentland, B.T.: Reconceptualizing organizational routines as a source of flexibility and change. Administrative Science Quarterly, 48(1), 94-118 (2003)
2. Feldman, M.S., Pentland, B.T., D'Adderio, L., Dittrich, K., Rerup, C., Seidl, D.: What Is Routine Dynamics?. In: Feldman, M.S., Pentland, B.T., D'Adderio, L., Dittrich, K., Rerup, C., Seidl, D. (eds.) Cambridge Handbook of Routine Dynamics. Cambridge University Press (2021)

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63783-4_27

3. Li, W., Badr, Y., Biennier, F.: Digital ecosystems: Challenges and prospects. In: MEDES '12: Proceedings of the International Conference on Management of Emergent Digital EcoSystems. ACM, New York (2012)
4. Deng, T., Robinson, W.N.: Changes in emergent software development routines: The moderation effects of routine diversity. International Journal of Information Management, 58 (2021)
5. Hayes, G.R., Lee, C.P., Dourish, P.: Organizational routines, innovation, and flexibility: The application of narrative networks to dynamic workflow. International Journal of Medical Informatics, 80(8), 161-177 (2011)
6. Pentland, B.T., Feldman, M.S.: Narrative networks: Patterns of technology and organization. Organization Science, 18(5), 781-795 (2007)
7. Howard-Grenville, J., Rerup, C.: A process perspective on organizational routines. In: Langley, A, Haridimos T. (eds.) The SAGE Handbook of Process Organization Studies. SAGE (2016)
8. Pentland, B.T., Liu, P., Kremser, W., Hærem, T.: The Dynamics of Drift in Digitized Processes. MIS Quarterly, 44(1) (2020)
9. Pentland, B. T., Hærem, T., Hillison, D.: Comparing organizational routines as recurrent patterns of action. Organization Studies, 31(7), 917-940 (2010)
10. Gaskin, J., Berente, N., Lyytinen, K., Yoo, Y.: Toward generalizable sociomaterial inquiry. MIS Quarterly, 38(3), 849-A812 (2014)
11. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825-2830 (2011)
12. Thomas, L. D., Tee, R.: Generativity: A systematic review and conceptual framework. International Journal of Management Reviews, 24(2), 255-278 (2022)
13. Sele, K., Grand, S.: Unpacking the dynamics of ecologies of routines: Mediators and their generative effects in routine interactions. Organization Science, 27(3), 722-738 (2016)
14. Sammon, D., Nagle, T., McAvoy, J. Analysing ISD performance using narrative networks, routines and mindfulness. Information and Software Technology, 56(5), 465-476 (2014)
15. Cohen, M. D.: Reading Dewey: Reflections on the study of routine. Organization Studies, 28(5), 773-786 (2007)
16. Pentland, B.T., Recker, J., Wolf, J.R., and Wyner, G.: Bringing context inside process research with digital trace data. Journal of the Association for Information Systems, 21(5), 5 (2020)
17. Bissyandé, T. F., Lo, D., Jiang, L., Réveillere, L., Klein, J., Le Traon, Y.: Got issues? Who cares about it? A large scale investigation of issue trackers from GitHub. In: IEEE 24th International Symposium on Software Reliability Engineering. IEEE, (2013)
18. Tsay, J., Dabbish, L., Herbsleb, J.: Let's talk about it: Evaluating contributions through discussion in GitHub. In: Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering. ACM, New York (2014)
19. Scikit-learn governance and decision-making on scikit-learn.org, https://scikit-learn.org/stable/governance.html#governance, last accessed 2024/04/26
20. Scikit-learn community code of conduct on scikit-learn.org, https://github.com/scikit-learn/scikit-learn/blob/main/CODE_OF_CONDUCT.md, last accessed 2024/04/26
21. GitHub Docs - Issues, https://docs.github.com/en/issues/tracking-your-work-with-issues/about-issues, last accessed 2024/04/26

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63783-4_27

22. GitHub Docs - Collaborating with pull requests, https://docs.github.com/en/pull-requests/collaborating-with-pull-requests, last accessed 2024/04/26
23. Abbott, A., Tsay, A.: Sequence analysis and optimal matching methods in sociology: Review and prospect. Sociological Methods and Research, 29(1), 3-33 (2000)
24. Rabiner, L.R.: A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE, 77(2), 257-286 (1989)
25. PyGithub v2.2.0 release on GitHub, https://github.com/PyGithub/PyGithub/, last accessed 2024/03/14
26. Vasilakis, C., Lecznarowicz, D., Lee, C.: Developing model requirements for patient flow simulation studies using the Unified Modelling Language (UML). Journal of Simulation, 3(3), 141-149 (2009)
27. Dobing, B., Parsons, J.: How UML is used. Communications of the ACM, 49(5), 109-113 (2006)
28. Winter, S.G.: Understanding dynamic capabilities. Strategic Management Journal, 24(10), 991-995 (2003)
29. Cohen, M.D. and Bacdayan, P.: Organizational routines are stored as procedural memory: Evidence from a laboratory study. Organization Science, 5(4), 554-568 (1994)
30. Hansson, M., Hærem, T., Pentland, B.T.: The effect of repertoire, routinization and enacted complexity: Explaining task performance through patterns of action. Organization Studies, 44(3), 473-496 (2021).

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63783-4_27