Reuse vs. Reusability of Software Supporting Business Processes

Hermann Kaindl[1(✉)], Roman Popp[1], Ralph Hoch[1], and Christian Zeidler[2]

[1] Institute of Computer Technology, TU Wien, Vienna, Austria
{kaindl,popp,hoch}@ict.tuwien.ac.at
[2] Adaptive GmbH, Vienna, Austria
zeidler@adaptive.at

Abstract. Reusing software is desirable, and so is reusing business processes. For reusing both in the course of developing software supporting business processes, an integration of related reuse approaches is necessary. Of course, such reuse is not for free and requires reusability of related artefacts, i.e., business process models and software parts supporting them. For successful reuse, of course, trade-offs with making artefacts reusable (or acquiring them) have to be beneficial.

In this paper, we present an integration of business process and software reuse and reusability (R&R). Based on it, we compare trade-offs between making reusable and reusing in the context of developing software supporting business processes. As a consequence, it should become easier to make rational judgments on whether and how to engage in R&R of such software.

Keywords: Reuse and reusability · Business process · Business software

1 Introduction

The context of this work is *reuse* in the course of creating and adapting software (SW) supporting *business processes*, where *reusability* depends on the explicit availability (and use) of a business process model (BPM). Reuse of software and of related BPMs together has the potential to increase efficiency and thus to reduce costs and time-to-market. However, the trade-offs with related investments into reusability need to be better understood.

More specifically, we focus on reuse based on repositories, as illustrated in Fig. 1. This approach integrates reuse of (similar) business processes and their adaptation for the case at hand (possibly also involving their composition) with reuse of related software parts (such as components or Web-services) and their adaptation. It requires repositories filled with reusable artefacts of both kinds, which can be efficiently looked-up for retrieval of (similar) artefacts as needed. This, in turn, requires some effort for making artefacts reusable. So, we discuss trade-offs between investments into reusability and related benefits for efficient software and process reuse.

G.M. Kapitsaki and E. Santana de Almeida (Eds.): ICSR 2016, LNCS 9679, pp. 138–145, 2016.
DOI: 10.1007/978-3-319-35122-3_10

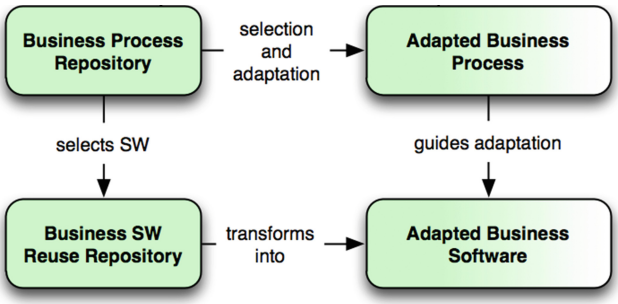

Fig. 1. Business software development with reuse from repositories

The remainder of this paper is organized in the following manner. First, we provide an overview of related work both on software and business process reuse and reusability. Then we explain an integration of software and business process reuse. For such a reuse approach, we compare trade-offs between making reusable and reusing in the context of developing software supporting business processes.

2 Related Work

Software reuse and reusability have a long tradition in general, see, e.g., [4], where Frakes and Terry reviewed, among other things, metrics and cost-benefit models. Rotaru and Dobre [14] studied the adaptability and composability of software components, both qualitatively and quantitatively (through metrics). Recently, Mohr [10] presented metrics for functional reusability of services based on their relevance. So, for software parts even quantitative measures related to their reusability are available. These could be used in the context of our approach for software reuse.

Reuse of business process models is the act of designing business processes by using existing process models. To this end, typically BPM repositories are employed. Requirements for such repositories from a stakeholders' perspective were defined in [16]. Elias and Johannesson [3] provided a survey on repositories for process models. A similar survey was carried out by Yan et al. [17]. Such repositories may serves as building blocks in the context of our approach for BPM reuse.

For retrieving relevant BPMs from such a repository, Dijkman et al. [2] described graph matching on business processes to search and find similar processes. Business process fragments may be reused during business process modeling by integration [9]. According to [1], business processes are compositions of sub-processes or process fragments. Both composability and variability are necessary for deploying a business process in an adapted way. This work may be used in the context of our approach for BPM retrieval and adaptation.

When both a BPM and related software parts are available, software supporting the modeled process may be directly driven by the BPM [11]. Based on this idea, we recently proposed a software architecture including a BPMN 2.0 engine and a model of business artefacts for aligning the architectures of the business

and its supporting software [5]. BPMs can be enriched at their enactment with additional artefact information for addressing certain usability problems of such software. We build on this previous work in our overall approach for integrating a BPM directly in the software.

3 Integrated Software and Business Process Reuse

Based on this previous work, integrated software and business process reuse is possible as illustrated in Fig. 2. Business Software Reuse as sketched at the bottom of the figure may happen with virtually any software reuse approach. The figure shows a simple case-based approach, where software cases are stored in a repository, selected using some similarity measure, and adapted for the case at hand. Even a single scenario was sufficient for finding useful software cases in [7]. Such an approach is also part of a feature-similarity model for product line engineering recently co-proposed by one of these authors [8].

Business Software Reuse is integrated in our approach with Business Process Reuse as sketched at the top of the figure. Also for such a reuse, different approaches are possible. Analogously to software reuse, the figure sketches the selection of a business process (more precisely a BPM) from a repository and its adaptation. According to [13], such a process adaptation can be an *adjustment* or a *refinement*. Both may be performed even automatically through model transformations specifying business rules (see also [12]). Model transformations have also been used for automated tailoring of a software process [6], but we consider this outside the scope of our approach as presented here.

Ideally, every BPM in the repository could be executed using the software artefacts in the repository. After a process adaptation, however, some part of the adapted BPM, e.g., a Task (as illustrated in green in Fig. 2), may *not* be executable by any piece of software in the repository. Then a related software adaptation will be necessary. It may have to be done manually, but model transformations could be employed as well.

4 Comparison of R&R Trade-offs

The R&R trade-offs in the context of software supporting business processes are between an initial investment to create *reusable* software or BPM artefacts, and the benefits from having either or both of them available for later *reuse*. We compare such trade-offs in three different scenarios that primarily differ in what is given for a development or change effort:

- *Software development from scratch*
  This is the extreme case where nothing would exist yet for being reused, not even software built for prior (similar) projects.
- *Software available, but neither BPMs nor repositories for reuse*
  This is a case where software exists, which has to be changed or may be informally used somehow for creating similar software. However, no investment

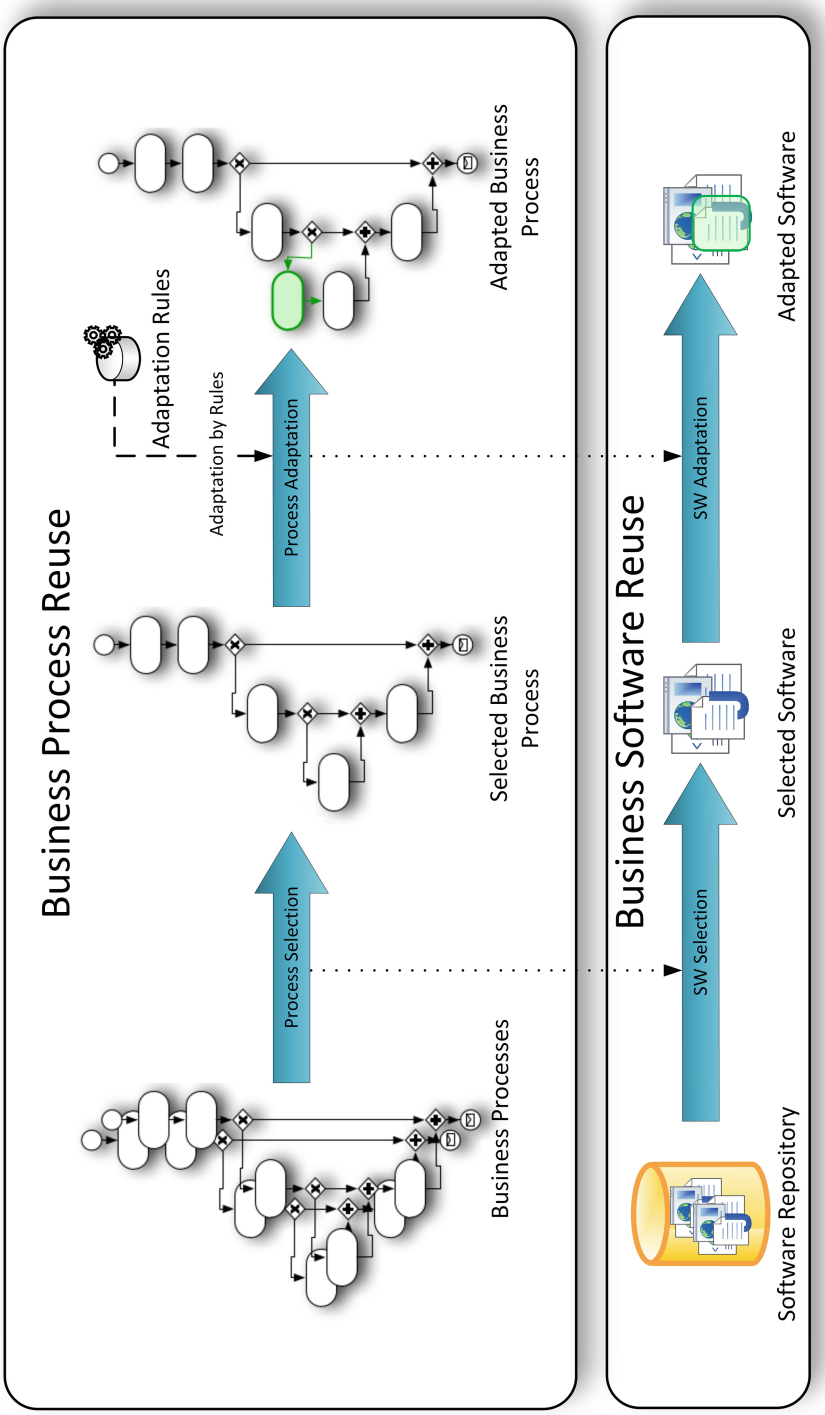

Fig. 2. Integrated business process and software reuse

into (systematically) making artefacts reusable has been done yet, neither for software nor BPM artefacts.
– *Repositories filled with reusable artefacts*
This is the other extreme case where investments have been made for creating both software and BPM repositories with reusable artefacts.

These scenarios are obviously on different levels of R&R maturity for software (see, e.g., [4]). However, they do not involve software artefacts and processes only but also related business process artefacts or processes dealing with them.

In Table 1, these scenarios are given in its rows. The columns contrast software development without any systematic reuse or reusability with the R&R approaches illustrated above in Fig. 2. In the third column, a software repository filled with reusable software artefacts is assumed to be available and used. In the fourth column, in addition, a related BPM repository filled with reusable BPMs is assumed to be available and used. "**MR**" indicates an investment through *making reusable*, while "**R**" stands for *reusing*.

Such a trade-off can obviously be in terms of some *cost* measure. As discussed below, however, investing some cost for MR may have a positive return by R in terms of *time*, e.g., time-to-market, i.e., in a different 'currency'. We also discuss

Table 1. A comparison of approaches to software development and change based on reuse and reusability

	Software development without R&R	Reuse with software repository	Reuse with BPM repository
Software development from scratch	Software development only	**MR**: repository has to be available (or created)	**MR**: repository has to be available (or created)
	Developers directly encode BP in source code	**MR**: enrich software parts with meta-data	**MR**: enrich BPMs with meta-data
	Alternative: executable BPM available or created	**MR**: organize software parts in repository	**MR**: organize BPMs in repository
Software available, but neither BPMs nor repositories for reuse	Source code has to be changed	*Same as in cell above*	*Same as in cell above*
	Depending on the architecture, more or less complicated	Possibly some reverse engineering	Possibly some reverse engineering
Repositories filled with reusable software artefacts and BPMs	—	**R**: possibility to retrieve software artefacts from repository for reuse	**R**: possibility to retrieve BPMs from repository for reuse
			R: possibility to retrieve related software artefacts from repository for reuse

positive and negative results in terms of *quality*. So, we discuss trade-offs with a triple (cost, time, quality), which was also inspired by [15].

Let us start with the scenario of software development from scratch. If it focuses on development only, then there is no investment into explicit reuse later. If there is no BPM available, developers directly encode the business process in the source code. However, if an executable PBM is explicitly given or created for the software supposed to support this process, this BPM may be directly included into a specific software architecture and drive the software at runtime (see, e.g., [5]). While this approach can be efficient, it reduces the flexibility of the software and may even entail usability problems. In terms of making such software reusable, investments should be made here to enter pieces of software such as components or Web services into a software repository. This requires that a repository is technically available or has to be created, and the artefact to be stored has to be enriched with meta-data and organized into the repository. In addition, the BPM should be made reusable as well by entering it into a repository, analogously to entering pieces of software. These investments are usually in terms of cost. While they will also take extra time, it can be spent in parallel to development projects.

When software is already available from previous projects, but neither BPMs nor repositories for reuse, then source code has to be added or changed directly. The difficulty of doing this will depend, e.g., on the software architecture. If an executable BPM drives the software, primarily adaptations of such a model will have to be made. Investments for making such software or BPMs reusable in repositories are basically the same as indicated above. When this is done only after several projects have already created software and models without making them reusable, then even some reverse engineering may have to be done additionally now.

For the scenario with repositories already filled with reusable artefacts, development will try to reuse as many as possible to make best use of them. Let us first have a brief look at the well-known case where software artefacts (only) are available for reuse in a repository. In general, it will be more efficient than software development from scratch, i.e., there will be a return of invest from MR for R in terms of cost. Actually, there should also be an improvement in time-to-market, where the investment by MR in terms of cost is paid back in terms of time. When software artefacts are often reused, it is well-known that they may become more mature, i.e., there may be a return of invest in terms of quality.

If an executable BPM drives the software, primarily adaptations of such a model will have to be made and, if they can be implemented completely by software parts from the repository, ideally no software developer will have to make any change to the source code. This requires a given framework for automatic execution of BPMs, however, with the possible downside of reduced flexibility and quality, e.g., of the user interface.

If, in addition, a repository full of BPMs is available, then they can be reused as well. In particular, BPMs may be found in the repository as needed, and two or more of these BPMs may be merged. If these BPMs are supported well by stored

software parts, then ideally not much new software will have to be created anew, in stark contrast to pure software development in such a case without reuse. The return of invest from MR on this level may also be in terms of cost, time and quality through R of BPMs, much as through indirect R of related software artefacts.

5 Conclusion

In this paper, we discuss trade-offs between reuse and reusability of software supporting business processes, depending on different development approaches with and without explicit business process models and corresponding repositories. This discussion is based on the literature and on previous work of these authors.

This work aims to contribute an improved understanding of these trade-offs for different development approaches for software supporting business processes. In particular, such trade-offs may arise with different currencies, e.g., cost vs. time-to-market, with effects on quality as well. We found an argument why efforts into making artefacts should be invested early, since otherwise even additional effort in some reverse engineering may arise. Overall, a reuse approach integrating both business processes and software artefacts appears to have a high potential.

Still, our comparison is based on qualitative assessments only. Based on already existing work on metrics especially in the context of software reuse and reusability, future work should investigate such trade-offs quantitatively as well.

Acknowledgment. Part of this research has been carried out in the ProREUSE project (No. 834167), funded by the Austrian FFG.

References

1. Angles, R., Ramadour, P., Cauvet, C., Rodier, S.: V-BPMI: A variability-oriented framework for web-based business processes modeling and implementation. In: 2013 IEEE Seventh International Conference on Research Challenges in Information Science (RCIS), pp. 1–11, May 2013
2. Dijkman, R., Dumas, M., García-Bañuelos, L.: Graph matching algorithms for business process model similarity search. In: Dayal, U., Eder, J., Koehler, J., Reijers, H.A. (eds.) BPM 2009. LNCS, vol. 5701, pp. 48–63. Springer, Heidelberg (2009). doi:10.1007/978-3-642-03848-8_5
3. Elias, M., Johannesson, P.: A survey of process model reuse repositories. In: Dua, S., Gangopadhyay, A., Thulasiraman, P., Straccia, U., Shepherd, M., Stein, B. (eds.) ICISTM 2012. CCIS, vol. 285, pp. 64–76. Springer, Heidelberg (2012). doi:10.1007/978-3-642-29166-1_6
4. Frakes, W., Terry, C.: Software reuse: metrics and models. ACM Comput. Surv. **28**(2), 415–435 (1996). doi:10.1145/234528.234531

5. Hoch, R., Kaindl, H., Popp, R., Zeidler, C.: Aligning architectures of business and software: software driven by business process models and its user interface. In: Proceedings of the 2016 49th Hawaii International Conference on System Sciences, HICSS 2016. IEEE Computer Society (2016)
6. Hurtado Alegría, J.A., Bastarrica, M.C., Quispe, A., Ochoa, S.F.: An MDE approach to software process tailoring. In: Proceedings of the 2011 International Conference on Software and Systems Process, ICSSP 2011, pp. 43–52, ACM, New York, NY, USA (2011). http://doi.acm.org/10.1145/1987875.1987885
7. Kaindl, H., Śmiałek, M., Nowakowski, W.: Case-based reuse with partial requirements specifications. In: 18th IEEE International Requirements Engineering Conference (RE 2010), pp. 399–400, IEEE, New York, NY, USA (2010)
8. Kaindl, H., Mannion, M.: A feature-similarity model for product line engineering. In: Schaefer, I., Stamelos, I. (eds.) ICSR 2015. LNCS, vol. 8919, pp. 34–41. Springer, Heidelberg (2014). doi:10.1007/978-3-319-14130-5_3
9. Markovic, I., Pereira, A.C.: Towards a formal framework for reuse in business process modeling. In: ter Hofstede, A., Benatallah, B., Paik, H.-Y. (eds.) BPM Workshops 2007. LNCS, vol. 4928, pp. 484–495. Springer, Heidelberg (2008). http://dl.acm.org/citation.cfm?id=1793714.1793769
10. Mohr, F.: A metric for functional reusability of services. In: Schaefer, I., Stamelos, I. (eds.) ICSR 2015. LNCS, vol. 8919, pp. 298–313. Springer, Heidelberg (2014). doi:10.1007/978-3-319-14130-5_21
11. Ouyang, C., Dumas, M., Van Der Aalst, W.M., Ter Hofstede, A.H., Mendling, J.: From business process models to process-oriented software systems. ACM Trans. Softw. Eng. Methodol. **19**(1), 2:1–2:37 (2009). doi:10.1145/1555392.1555395
12. Popp, R., Kaindl, H.: Automated adaptation of business process models through model transformations specifying business rules. In: Nurcan, S., Pimenidis, E., Pastor, O., Vassiliou, Y. (eds.) Joint Proceedings of the CAiSE 2014 Forum and CAiSE 2014 Doctoral Consortium Co-located with the 26th International Conference on Advanced Information Systems Engineering (CAiSE 2014), Thessaloniki, Greece, CEUR Workshop Proceedings, vol. 1164, pp. 65–72. CEUR-WS.org, 18–20 June 2014. http://ceur-ws.org/Vol-1164/PaperVision09.pdf
13. Popp, R., Kaindl, H.: Automated refinement of business processes through model transformations specifying business rules. In: 9th IEEE International Conference on Research Challenges in Information Science, RCIS 2015, Athens, Greece, pp. 327–333. IEEE, 13–15 May 2015. http://dx.doi.org/10.1109/RCIS.2015.7128893
14. Rotaru, O., Dobre, M.: Reusability metrics for software components. In: The 3rd ACS/IEEE International Conference on Computer Systems and Applications, p. 24 (2005)
15. Sametinger, J.: Software Engineering with Reusable Components. Springer, New York (1997)
16. Shahzad, K., Elias, M., Johannesson, P.: Requirements for a business process model repository: a stakeholders' perspective. In: Abramowicz, W., Tolksdorf, R. (eds.) BIS 2010. LNBIP, vol. 47, pp. 158–170. Springer, Heidelberg (2010). doi:10.1007/978-3-642-12814-1_14
17. Yan, Z., Dijkman, R., Grefen, P.: Business process model repositories - framework and survey. Inf. Softw. Technol. **54**(4), 380–395 (2012). http://www.sciencedirect.com/science/article/pii/S0950584911002291