Do We Need a Refined Choreography Notion?

Andreas Schönberger

Distributed and Mobile Systems Group,
University of Bamberg
Bamberg, Germany
andreas.schoenberger@uni-bamberg.de

Abstract. Since the term *choreography* for capturing the publicly observable message exchanges between integration partners was coined, choreography technology evolved significantly. Today, the diversity of choreography languages is high. Up to now, choreography languages have been categorized by distinguishing between implementation specific and implementation independent choreographies as well as interaction and interconnection choreographies.
In this work, we review important characteristics of choreography technologies to find out whether a refined choreography notion is needed. The fact that choreography classes that are almost orthogonal to existing categorizations as well as several selective choreography characteristics can be identified suggests this need.

Keywords: B2Bi Choreography, Services Choreography, Conceptual Choreography

1 Introduction

In 2003, Chris Peltz coined the terms *Web Services Choreography and Orchestration* by distinguishing between *tracking the messages between* integration partners and the *executable local processes* of individual integration partners (cf. [11]). While Peltz tied the notion of *Choreography* to Web Services, today, there are a number of Web Services agnostic choreography languages such as UMM [18], ebXML BPSS (ebBP) [9] or Let's Dance [22]. However, capturing publicly visible messages between entities has remained as common characteristic of choreography languages.
Decker, Kopp and Barros [3] developed a categorization of choreographies based on two pivotal properties of choreography languages. First, they distinguish between *interconnection* choreographies that focus on the local send and receive actions of individual partners as well as the interconnection of corresponding send/receive actions and *interaction* choreographies that treat corresponding send and receive events as atomic actions and define sequences of these actions. Second, they distinguish between *implementation specific* choreographies that capture implementation level concepts like communication technology (say, Web Services) and *implementation independent* choreographies that are agnostic to

those concepts.
While this categorization for sure is pivotal it still captures languages with considerable differences in the same category. For example, Let's Dance and ebBP can both be characterized as implementation-independent interaction choreographies. However, ebBP targets at specifying the business document exchanges between enterprises while Let's Dance targets at supporting service interaction patterns [2] with a visual choreography language. Although these two goals overlap, they result in substantially different concepts. ebBP provides support for referencing existing business document libraries as provided by RosettaNet[1] and for specifying security and reliability requirements. Also, it assumes a protocol consisting of several message exchanges for implementing a business document exchange. Let's Dance, in turn, offers functionality for analyzing such protocols and provides a rich set of features for modeling service interactions.
These differences are a first hint that a refined choreography notion may be needed. This paper is dedicated to the investigation of that need. In Sect. 2, the analysis framework for conceptual modeling languages put forward by Wand and Weber [21] is used to derive *B2Bi/Services/Conceptual Choreographies* as distinct choreography classes that are largely orthogonal to the categorization presented in [3]. In Sect. 3, we identify 15 criteria that discriminate well between choreography categories. From these two results, we conclude that a refined choreography notion indeed is needed for helping practitioners and researchers in choosing a choreography language that fits their needs. Section 4 briefly discusses related work and Sect. 5 concludes and points out directions for future work.

2 Choreography Classes

In [21], Wand and Weber present an analysis framework for conceptual modeling. For comparing languages, the framework components *task factors* capturing the purpose of using a language as well as *modeling grammar* capturing the constructs and rules for creating models are relevant.
While choreography languages may lend itself to a variety of different purposes, it is striking that almost all choreography languages and approaches underline their relevance for Business-to-Business integration (B2Bi). Publications such as [7] and [15] that analyze the development phases of B2Bi and therefore are suited to identify *task factors* reveal that choreography technologies typically are used to fill the semantic gap between business process models (BPM) and orchestration models (OM). This can be done by refining BPMs or by abstracting OM concepts. For example, BPEL4Chor [4] reuses a considerable part of WS-BPEL concepts which corresponds to *abstracting OM concepts*. Conversely, ebBP uses so-called BusinessTransactions to specify requirements of message exchanges which corresponds to *refining the BPM layer*. Finally, for some choreography languages it is not easily decidable whether they are semantically more close to the BPM layer or to the OM layer. For example, IOWF-Nets [19] capture choreographies as interconnected Petri Nets. This resembles the concept of composing

[1] `http://www.rosettanet.org/`

a choreography by connecting orchestrations and therefore seems to imply a close relationship to the OM layer. However, the BPM layer may contain partner-local models as well and as IOWF-Nets do not have technology specific concepts, they could potentially be used for analyzing the BPM layer itself too.
These differences are also reflected in the core building blocks of the various choreography languages (cf. *modeling grammar* [21]). In BPEL4Chor, *communication activities* are used to capture the send and receive events of the individual partners. The WSDL interaction styles 'one-way' and 'request/response' are adopted to allow for *"higher similarity between participant behavior descriptions and orchestrations"* ([5], section 4.2). So, although BPEL4Chor is defined such that it does not *technically depend* on WSDL (by removing the *partnerLink*, *portType*, and *operation* attributes from BPEL communication activities), it can be concluded that BPEL4Chor is *designed for* services based interactions. In ebBP, a BusinessTransaction represents a B2Bi domain specific configuration of a business document exchange with B2Bi parameters for a lower-level execution protocol. Finally, the core building blocks of languages like IOWF-Nets or Let's Dance neither rely on Web Services concepts nor define B2Bi domain concepts.
On the basis of this analysis of *tasks factors* and *modeling grammar* as supposed by [21], at least three different classes of choreography languages (largely orthogonal to the categorization in [3]) can be identified:

B2Bi Choreographies that offer B2Bi specific concepts like configurable BusinessTransactions and which semantically are close to BPM models.
Services Choreographies that offer Web Services technology specific concepts and are close to the orchestration layer.
Conceptual choreographies that offer concepts driven by the purpose of analysis and may be used to complement/analyze the BPM layer as well as the OM layer.

3 Selective Criteria

While the last section shows that major different choreography classes can be distinguished by *task factors* and *modeling grammar elements*, this section identifies criteria that promise to discriminate between different categories of choreographies, i.e., which have the same value for some choreography languages but not for all. The criteria have been collected by leveraging two publications that postulate requirements for the important classes of services choreographies [5] and B2Bi choreographies [12] respectively and by reviewing design drivers of various choreography related publications, in particular [1,2,3,4,6,11,16,22].
The resulting criteria then have been filtered by removing criteria that do not discriminate well or can be derived almost functionally from other criteria. For evaluating selectiveness between choreography categories, the following representatives from the aforementioned choreography classes have been chosen that also represent all fields of the choreography categorization matrix from [3]:
IOWF-Nets [19] represent conceptual choreographies as well as implementation-

independent interconnection choreographies. Let's Dance [22] represents conceptual choreographies as well as implementation-independent interaction choreographies. ebBP [9] represents B2Bi choreographies as well as implementation-independent interaction choreographies. WS-CDL [20] represents services choreographies as well as implementation-specific interaction choreographies. Finally, BPEL4Chor [4] represents services choreographies as well as implementation specific interconnection choreographies.
While the identified criteria and the selected choreography languages for sure do not cover all aspects of choreography technology, the results described in Table 1 nonetheless demonstrate that there are several criteria that discriminate well between existing categories of choreographies. This, in turn, proves evidence for the fact that a refined choreography notion is needed. Note that the criteria of Table 1 are not intended as comparison framework for comparing in detail *choreography languages* of the same choreography class, but rather for distinguishing between *choreography categories*. While a single-author qualitative study (which always is biased to some extent) like the one at hand is sufficient for identifying the need for a refined choreography taxonomy, the development of a comprehensive choreography taxonomy calls for a joint effort of choreography researchers. Below, the individual criteria are presented:

1 Implementation Independence. Corresponds to the implementation specific/independent distinction as described in Sect. 1.
2 Communication Focus. Corresponds to the interconnection/interaction distinction as described in Sect. 1.
3 Core Design Driver. The core design driver of a choreography language can be inferred from its core building blocks and design rules and is frequently stated in related publications. For Let's Dance as well as BPEL4Chor, support for *Service Interaction Patterns* [2] is explicitly postulated as design driver in [22] and [5] respectively. For ebBP, *composition of BusinessTransactions* is the core design driver while *composition of interactions* apparently drove the design of WS-CDL. Finally, IOWF-Nets [19] result from extending the reach of WF-Nets to inter-organizational systems and therefore can be considered to be *formalism driven*. Obviously, choosing a core design driver does not uniquely determine language design.
4 Decomposability. Recursive decomposition of models is a frequent language design goal and is explicitly postulated in [12] and [22]. This criterion is represented as a yes/no value.
5 Distinction between participants and participant types. This criterion fosters *Service Interaction Patterns* support because it enables multiple instances of the same type of partner/service (cf. [22], [5]). We distinguish between *explicit* support and *no* support.
6 Domain. This criterion distinguishes between choreography languages that focus on an application domain such as *B2Bi* and *general* purpose languages.
7 Error Handling. This yes/no criterion is identified in both, [5] and [12], and is valued depending on the existence of explicit error handling concepts or

techniques.

8 Executability. Whether a model can be executed or not is a property of the particular model (or an according approach) and not of the language a model is composed from. However, there a different ways that choreography languages can be used. B2Bi choreographies are frequently used to just create a *cartography* of the types and scenarios of business document exchanges without intending to derive an implementation in a (semi-)automated manner. However, that does not exclude automated derivation of the control flow (cf. [13,16]). Moreover, a choreography may be used as a *blueprint* for identifying important parts of an implementation in a semi-automated manner [5]. Accordingly, *cartography*, *executable* and *blueprint* are possible values of this criterion.

9 Integration of Structural and Behavioral Views. Supporting the behavioral view on a system in the sense of constraining admissible message exchange sequences is a natural quality of choreography languages. However, some choreography languages additionally describe structural aspects such as the topology of services. Hence, *behavioral* and *integrated* can be assigned as values for this criterion.

10 Link Mobility. Some integration scenarios require the capability to pass on the endpoint of a communication partner to a third party. This capability, frequently denoted as link mobility, has been identified in [20] and [5] and is well-known from the π-calculus. This criterion is valued *explicit* or *no* support depending on the existence of dedicated link mobility constructs.

11 Processing Signals. For notifying a business document sender about the processability of the document, ebBP offers so-called Receipt-/AcceptanceAcknow -ledgements as *processing signals*. Identical concepts are also available in UMM [18] and the *Business Choreography Language* [23]. Note that processing signals are not first-class business messages as their existence depends on business document exchanges. This criterion is valued on a yes/no basis.

12 Protocol Abstraction. While Let's Dance or BPEL4Chor assume a one-to-one correspondence between a message exchange at the choreography level and its corresponding message exchange on the orchestration level, ebBP assumes a full communication protocol for implementing a single choreography exchange, i.e., a BusinessTransaction. *Protocol abstraction* captures the fact that a full protocol may be represented by a single exchange at the choreography level and accordingly is assigned a yes/no value.

13 Runtime Determination of Participants. [2] and [5] postulate the requirement for choreography languages to be able to leave the number of participant instances unspecified until runtime. This is different from *Link Mobility* as it does not necessarily require passing on communication endpoints. This criterion is valued *explicit* if the choreography language has explicit constructs for capturing that or *no* support otherwise.

14 Standardization. Although not a first-citizen property of languages, *standardization* of a language affects the selection of available constructs as well as its amenability to change. For industry and academia, whether or not a choreography language is a *standard* or a *research* prototype makes an important difference.

Table 1. Selective Criteria for Comparing Choreography Languages

Criterion	Let's Dance	ebBP	WS-CDL	BPEL4Chor	IOWF-Nets
1 Implementation Independence	independent	independent	specific	specific	independent
2 Communication Focus	interaction	interaction	interaction	inter-connection	inter-connection
3 Core Design Driver	interaction patterns	Business-Transaction composition	interaction composition	interaction patterns	formalism driven
4 Decomposability	yes	yes	yes	no	no
5 Distinction between participants and participant types	explicit	no	no	explicit	no
6 Domain	general	B2Bi	general	general	general
7 Error Handling	no	yes	yes	yes	no
8 Executability	cartography/ blueprint	cartography/ executable	cartography/ blueprint	blueprint	blueprint
9 Integration of Structural and Behavioral Views	behavioral	behavioral	integrated	integrated	integrated
10 Link Mobility	explicit	no	explicit	explicit	no
11 Processing Signals	no	yes	no	no	no
12 Protocol Abstraction	no	yes	no	no	no
13 Runtime Determination of Participants	explicit	no	no	explicit	no
14 Standardization	research	standard	standard	research	research
15 Transaction Safety	no	choice	choice	no	no

15 Transaction Safety. While ebBP by default assumes that business document exchanges are performed in a transactional manner, languages such as IOWF-Nets deliberately choose to separate sending messages from receiving messages and do not assume transaction safety. This is influenced by the fact that transaction safety for simple one-way or request/response interactions can easily be implemented using reliable messaging or distributed transaction features of the underlying middleware. Conversely, B2Bi business document exchanges at the choreography level may represent complex multi-message exchanges at the orchestration level [17,14] that reflect whether or not transaction safety is required explicitly. In [8], so-called *choreography spheres* are proposed to guarantee transaction safety for sets of choreography-level activities using advanced transaction features of the underlying BPEL engines. That approach can be applied on top of BPEL4Chor which does not provide built-in transaction safety support.

4 Related Work

In [11], the distinction between choreography and orchestration first was described. However, an analysis of different choreography classes is not provided. In [3], a categorization based on implementation independence and the interconnection/interaction dichotomy was proposed. The work at hand shows that

this categorization can be extended and complemented. In [24], requirements and language concepts for modeling cross-organizational business processes are identified. However, the focus is not put on choreographies in particular. Instead, the BPM layer and the OM layer are considered as well. Consequently, only 1 out of 7 requirements and 2 out of 7 language concepts distinguish well between categories of choreographies (evaluated for the languages used in table 1).
Finally, there is an abundance of publications postulating requirements for languages for particular purposes such as [5] for supporting service interaction patterns or [12] for B2Bi. However, these requirement sets are aligned with the design purpose and not with the intent to distinguish between choreography categories.

5 Conclusion and Future Work

This work contributes to choreography research by extending and complementing existing choreography categorizations which implies the need for a refined choreography notion. Researchers and practitioners benefit from the identification of choreography classes and from a number of selective criteria that discriminate well between choreography categories. Note that these criteria are unlikely to be '*met*' by a single language. In so far, they are also design options for choosing a choreography language. Also, the identified classes and criteria can be used by new choreography technologies such as the BPMN 2.0 choreographies ([10], section 11) to clarify its scope.
While this work proves the need for a refined choreography notion the identification of choreography classes and criteria is not complete. In so far, the results of this paper also call for a *joint* effort on extending choreography taxonomies.

References

1. A. Barros, G. Decker, and M. Dumas. Multi-staged and multi-viewpoint service choreography modelling. In *Proc. of the Workshop on Software Engineering Methods for Service Oriented Architecture (SEMSOA)*, volume 244 of *CEUR-WS*, May 2007.
2. A. Barros, M. Dumas, and A. H. M. T. Hofstede. Service interaction patterns. In *Proceedings of the 3rd International Conference on Business Process Management (BPM), Nancy, France*, pages 302–318. Springer Verlag, 2005.
3. G. Decker, O. Kopp, and A. Barros. An introduction to service choreographies. *Information Technology*, 50(2):122–127, 2008.
4. G. Decker, O. Kopp, F. Leymann, and M. Weske. BPEL4Chor: Extending BPEL for modeling choreographies. In *Proc. of the 2007 IEEE Int. Conf. on Web Services (ICWS), July 9-13, 2007, Salt Lake City, Utah, USA*, pages 296–303, 2007.
5. G. Decker, O. Kopp, F. Leymann, and M. Weske. Interacting services: From specification to execution. *Data & Knowledge Engineering*, 68(10):946 – 972, 2009.
6. R. Dijkman and M. Dumas. Service-oriented Design: A Multi-viewpoint Approach. *International Journal of Cooperative Information Systems*, 13(4):337–368, 2004.
7. J. Dorn, C. Grün, H. Werthner, and M. Zapletal. A survey of B2B methodologies and technologies: From business models towards deployment artifacts. In *HICSS 2007: Proc. of the 40th Hawaii Int. Conf. on System Sciences, Hawaii*. IEEE.

8. O. Kopp, M. Wieland, and F. Leymann. Towards Choreography Transactions. In *Proc. of the 1st Central-European Workshop on Services and their Composition, ZEUS 2009, Stuttgart, Germany, March 2–3, 2009*, volume 438 of *CEUR-WS*, pages 49–54, 2009.
9. OASIS. *ebXML Business Process Specification Schema Technical Specification.* OASIS, 2.0.4 edition, December 2006.
10. OMG. *Business Process Model and Notation, v2.0.* OMG, January 2011.
11. C. Peltz. Web services orchestration and choreography. *Computer*, 36(10):46–52, 2003.
12. A. Schönberger. The CHORCH B2Bi approach: Performing ebBP choreographies as distributed BPEL orchestrations. In *Proc. of the 6th World Congress on Services 2010 (SERVICES 2010), Miami, Florida, USA*. IEEE, July 2010.
13. A. Schönberger, C. Pflügler, and G. Wirtz. Translating shared state based ebXML BPSS models to WS-BPEL. *International Journal of Business Intelligence and Data Mining*, 5(4):398 – 442, 2010.
14. A. Schönberger and G. Wirtz. Using Webservice Choreography and Orchestration Perspectives to Model and Evaluate B2B Interactions. In *The 2006 Int. Conf. on Software Engineering Research and Practice (SERP'06), Las Vegas, USA*, 2006.
15. A. Schönberger and G. Wirtz. Taxonomy on consistency requirements in the business process integration context. In *Proc. of 2008 Conf. on Software Engineering and Knowledge Engineering (SEKE)*, Redwood City, California, USA, July 2008.
16. A. Schönberger and G. Wirtz. Towards executing ebBP-Reg B2Bi choreographies. In *Proceedings of the 12th IEEE Conference on Commerce and Enterprise Computing (CEC'10), Shanghai, China.* IEEE, November 10-12 2010.
17. A. Schönberger, G. Wirtz, C. Huemer, and M. Zapletal. A composable, QoS-aware and web services-based execution model for ebXML BPSS businesstransactions. In *Proceedings of the 6th 2010 World Congress on Services (SERVICES2010), Fourth International Workshop on Web Services and Cloud Services Testing (WS-CS-Testing 2010), Miami, Florida, USA*. IEEE, July 2010.
18. UN/CEFACT. *UN/CEFACT's Modeling Methodology (UMM): UMM Meta Model - Foundation Module Version 1.0.* UN/CEFACT, 1.0 edition, 10 2006.
19. W. M. P. van der Aalst and M. Weske. The P2P approach to interorganizational workflows. In *CAiSE '01: Proceedings of the 13th International Conference on Advanced Information Systems Engineering*, pages 140–156, London, UK, 2001.
20. W3C. *Web Services Choreography Description Language.* W3C, 1.0 edition, November 2005.
21. Y. Wand and R. Weber. Research commentary: Information systems and conceptual modeling–a research agenda. *Info. Sys. Research*, 13(4):363–376, 2002.
22. J. M. Zaha, A. P. Barros, M. Dumas, and A. H. M. ter Hofstede. Let's Dance: A language for service behavior modeling. In *Proc. of the 14th Int. Conf on cooperative information systems (CoopIS'06)*, Montpellier, France, 10 2006.
23. M. Zapletal, T. Motal, and H. Werthner. The business choreography language (BCL) - a domain-specific language for global choreographies. In *Proc. of the 5th 2009 World Congress on Services (SERVICES 2009 PART II), Bangalore, India.* IEEE, September 2009.
24. J. Ziemann, T. Matheis, and J. Freiheit. Modelling of cross-organizational business processes. In *Proc of the 2nd Int. Workshop on Enterprise Modelling and Information Systems Architectures (EMISA'07), St. Goar, Germany, October 8-9, 2007*, pages 87–100.