Chapter 15
Transforming Disciplined IT Functions:
Guidelines for DevOps Integration

Anna Wiedemann
Neu-Ulm University of Applied Sciences, Germany

Manuel Wiesche
https://orcid.org/0000-0003-0401-287X
Technical University of Dortmund, Germany

Heiko Gewald
https://orcid.org/0000-0003-2107-2217
Neu-Ulm University of Applied Sciences, Germany

Helmut Krcmar
https://orcid.org/0000-0002-2754-8493
Technical University of Munich, Germany

ABSTRACT

In today's fast-changing environment, many organizations are applying the DevOps (Development and Operations) concept to transform their IT functions and establish cross-functional IT teams to deliver software services quickly, reliably, and safely with end-to-end responsibility. The results of an empirical study on which this chapter is based presents platform-oriented, application-oriented, and mobile-oriented DevOps setups, outlining areas of potential collaboration between these DevOps setups and the importance of aligning the aims of development (process agility) and operations (process rigor). Based on the study, six indicators of successful DevOps integration formulated as recommendations for successful IT function transformation were identified.

DOI: 10.4018/978-1-7998-4165-4.ch015

Copyright © 2021, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

INTRODUCTION

In response to the many opportunities and challenges of this digital era, many large companies have been rethinking the organizational setups of their IT functions. Digital innovators such as Google and Amazon regularly release information about the newest trends in technology and serve as role models for other companies. Furthermore, large organizations manage their departments as organizational silos in which development and operations employees tend to be highly specialized. As a result of the silo setup, strict managerial hierarchies and clearly defined work environments often make communication across IT subunits rare. A key task of IT departments is to provide new software while ensuring smooth operations and functionality. In the silo setup, software updates are typically planned, constructed and executed as waterfall-style command-and-control projects (Dery, Sebastian, & van der Meulen, 2017), new software functionalities are seldom provided and large-scale error-prone releases are often executed on the weekend to avoid extensive system outages during peak business times.

As companies adopt enhanced development processes to make projects successful, it has become clear that providing complete service lifecycle support is most efficient when operations and development activities are combined. Developing capabilities and collaboration across internal IT subunits is necessary for both clear intra-organizational communication and improved performance (Dhaliwal, Onita, Poston, & Zhang, 2011). As organizations realize that silo IT functions are less able to leverage digitalization to maximize business value, they are more frequently integrating cross-functional IT teams in their IT departments to ensure fast and stable IT services (Wiedemann, 2017). The concept of DevOps is widely used to structure cross-functional teams with end-to-end responsibility for one or more IT services (Debois, 2011). In fact, a 2017 global conducted DevOps benchmark study by the market research institute Forrester presents that more than 50% of the participating organizations have implemented DevOps by end of 2017 (Stroud, 2017).

DevOps combines agility and rigor in cross-functional teams. Agility is defined as a method that enables rapid reaction to changing user requirement in an iterative way (Hemon, Monnier-Senicourt, & Rowe, 2018). The architecture of established IT departments often yields monolithic IT system with tightly coupled centralized application systems and bulky, often rigid legacy systems. Although this structure worked very well for a long time, progressive organizations following DevOps principles are striving for greater agility. For example, by creating internal IT ecosystems with loosely coupled IT infrastructures like microservices for managing frontend systems that run on digital service platforms serving as back-end systems (Ross et al., 2016). Process rigor is defined as the stringent enforcement of and adherence to compliance and formal, structured IS processes. Process rigor builds upon formal, clearly defined, and detailed explanations of tasks, roles, activities, work products, measures, and methods related to IS software processes. An example of a plan-driven software development approach based on process rigor is the Capability Maturity Model Integrated (CMMI) (Lee, DeLone, & Espinosa, 2010). One way to achieve process rigor in software operation is by implementing the highly formalized Information Technology Infrastructure Library (ITIL)[1].

Within DevOps teams, people from software development and software operations work together to enhance the speed of delivery of new software features. This cross-departmental cooperation helps to solve communication barriers and shorten communication routes across departments. Furthermore, the team members develop a broader understanding of the software delivery lifecycle and an appreciation of what everyone is doing to ensure successful delivery of the service. This chapter presents the results of a qualitative research study based on 80 interviews with people experienced in DevOps setups and

12 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the publisher's webpage:
www.igi-global.com/chapter/transforming-disciplined-it-functions/259183

Related Content

Using Non-Intrusive Environmental Sensing for ADLS Recognition in One-Person Household
Long Niu, Sachio Saikiand Masahide Nakamura (2018). *International Journal of Software Innovation (pp. 16-29).*
www.irma-international.org/article/using-non-intrusive-environmental-sensing-for-adls-recognition-in-one-person-household/210452

MADES FP7 EU Project: Effective High Level SysML/MARTE Methodology for Real-Time and Embedded Avionics Systems
Alessandra Bagnato, Imran Quadri, Etienne Brosse, Andrey Sadovykh, Leandro Soares Indrusiak, Richard Paige, Neil Audsley, Ian Gray, Dimitrios S. Kolovos, Nicholas Matragkas, Matteo Rossi, Luciano Baresi, Matteo Carlo Crippa, Stefano Genolini, Scott Hansenand Gundula Meisel-Blohm (2014). *Handbook of Research on Embedded Systems Design (pp. 181-208).*
www.irma-international.org/chapter/mades-fp7-eu-project/116110

Application Security for Mobile Devices
Gabriele Costa, Aliaksandr Lazouski, Fabio Martinelliand Paolo Mori (2015). *Handbook of Research on Innovations in Systems and Software Engineering (pp. 562-588).*
www.irma-international.org/chapter/application-security-for-mobile-devices/117941

Individual Improvisation in Information Systems Development
Massimo Magni, Bernardino Proveraand Luigi Prosperpio (2009). *Handbook of Research on Modern Systems Analysis and Design Technologies and Applications (pp. 105-118).*
www.irma-international.org/chapter/individual-improvisation-information-systems-development/21064

Anticipating Requirements Changes-Using Futurology in Requirements Elicitation
João Pimentel, Emanuel Santos, Jaelson Castroand Xavier Franch (2012). *International Journal of Information System Modeling and Design (pp. 89-111).*
www.irma-international.org/article/anticipating-requirements-changes-using-futurology/65563