

Chapter 8

Disciplined Teams vs. Agile Teams: Differences and Similarities in Software Development

Antonio Alexandre Moura Costa

Federal Institute of Paraiba, Brazil

Felipe Barbosa Araújo Ramos

 <https://orcid.org/0000-0002-0937-811X>

Federal Institute of Paraiba, Brazil

Dalton Cézane Gomes Valadares

Federal University of Campina Grande, Brazil & Federal Institute of Pernambuco, Brazil

Danyllo Wagner Albuquerque

Federal University of Campina Grande, Brazil

Emanuel Dantas Filho

Federal University of Campina Grande, Brazil

Alexandre Braga Gomes

Federal University of Campina Grande, Brazil

Mirko Barbosa Perkusich

VIRTUS, Brazil

Hyggo Oliveira de Almeida

Federal University of Campina Grande, Brazil

ABSTRACT

Software development has been considered a socio-technical activity over the past decades. Particularly, in the case of software engineering, the necessity to communicate effectively with stakeholders and team

DOI: 10.4018/978-1-7998-4165-4.ch008

members has been progressively emphasized. Human resources play a critical role in the success of software projects. Many techniques, methods, tools, models, and methodologies have been proposed, applied, and improved in order to help and ease the management of the software development process. Regardless of the software development methodology adopted, delivering a quality product in a predictable, efficient, and responsive manner is the objective for every team. Disciplined and Agile teams have different characteristics, but also share common aspects when working to accomplish their goals. The main motivation of this chapter is to present the differences and similarities of both teams in the context of software development.

INTRODUCTION

Delivering high-quality products in time and without budget overrun is still a significant struggle for most software organizations. Among the most common reasons are inaccurate estimates of needed resources, unmanaged risks, sloppy development practices, poor project management, commercial pressures, among others (Charette, 2005; Macnab & Doctolero, 2019). Software projects failures have a major negative impact on the organizations and can deeply compromise its future.

Projects continue to proliferate in society today, in both the public and private sectors of the economy. Investments in projects number in the trillions of dollars annually. Just as ubiquitous as these projects, unfortunately, are their significant failure rates. The CHAOS reports have identified the current state of project success rates across organizations, noting that in spite of much higher visibility and importance placed on project performance, failure rates have remained high and relatively stable across over a decade of research. (Serrador & Pinto, 2015)

When a software project fails, it jeopardizes an organization's prospects. If the failure is large enough, it can steal the company's entire future. In one stellar meltdown, a poorly implemented resource planning system led FoxMeyer Drug Co., a \$5 billion wholesale drug distribution company in Carrollton, Texas, to plummet into bankruptcy in 1996. (Charette, 2005)

For a long time, Disciplined approaches had been used to increase project success. These approaches, also known as plan-driven or heavyweight methodologies, are conducted in a linear way, whereas process activities of specification, development, validation, and evolution must be performed in sequential order, which means that an activity must be completed before the next one begins. Due to the heavy aspect of the Disciplined approaches, several consultants have developed methodologies and practices with more emphasis on people, collaboration, customer interaction, and working software, rather than on processes, tools, documentation, and plans. These approaches, known as Agile methods, have been gained popularity over the past years, especially after the advent of the Agile Manifesto in 2001 (Fowler & Highsmith, 2001). Unlike Discipline approaches such as the Waterfall model, the Agile ones promote continuous iteration of development and testing throughout the software development lifecycle.

14 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/disciplined-teams-vs-agile-teams/259176

Related Content

Cloud Computing Virtual Machine Workload Prediction Method Based on Variational Autoencoder

Fargana J. Abdullayeva (2021). *International Journal of Systems and Software Security and Protection* (pp. 33-45).

www.irma-international.org/article/cloud-computing-virtual-machine-workload-prediction-method-based-on-variational-autoencoder/284559

Enforcing ASTD Access-Control Policies with WS-BPEL Processes in SOA Environments

Michel Embe Jiague, Marc Frappier, Frédéric Gervais, Régine Laleau and Richard St-Denis (2013). *Mobile and Web Innovations in Systems and Service-Oriented Engineering* (pp. 252-273).

www.irma-international.org/chapter/enforcing-astd-access-control-policies/72001

Evaluation of Dynamic Analysis Tools for Software Security

Michael Lescisin and Qusay H. Mahmoud (2018). *International Journal of Systems and Software Security and Protection* (pp. 34-59).

www.irma-international.org/article/evaluation-of-dynamic-analysis-tools-for-software-security/221930

Designing Secure and Privacy-Aware Information Systems

Christos Kalloniatis, Argyri Pattakou, Evangelia Kavakli and Stefanos Gritzalis (2017). *International Journal of Secure Software Engineering* (pp. 1-25).

www.irma-international.org/article/designing-secure-and-privacy-aware-information-systems/190419

Fuzzy Ontology for Requirements Determination and Documentation During Software Development

Priti Srinivas Sajja and Rajendra A. Akerkar (2022). *Research Anthology on Agile Software, Software Development, and Testing* (pp. 726-745).

www.irma-international.org/chapter/fuzzy-ontology-for-requirements-determination-and-documentation-during-software-development/294492