References

1. Voigt, S., Kemper, T., Riedlinger, T., Freire, S., d'Andrimont, R., Tonolo, F. G., Lemoine, F., & Lang, S. (2015). Earth observation and GIS to support humanitarian operations in refugee/IDP camps. ResearchGate.

2. Cho, S., Lee, J., Hwang, Y., & Kwon, S. (2022). Humanitarian logistics challenges in disaster relief operations: A humanitarian organisations' perspective. ResearchGate.

3. Dziuba, S., Bulat, A., Koriashkina, L., & Blyuss, B. (2023). Discrete-Continuous Model of the Optimal Location Problem for the Emergency Logistics System. SSRN Electronic Journal. https://doi.org/10.2139/ssrn.4401341

4. Google Maps Platform. (n.d.). Documentation for nearby search. Retrieved from https://developers.google.com/maps/documentation/places/web-service/nearby-search

5. Google Maps Platform. (n.d.). Documentation for reverse geocoding. Retrieved from https://developers.google.com/maps/documentation/geocoding/requests-reverse-geocoding

UDC 658.5

AGILE FRAMEWORKS. NEXUS FRAMEWORKS FOR SCALING SCRUM

Prof. Dr. Glöckle Herbert in Reutlingen University
Anastasiia Maliienko – maliienko.a.a@nmu.one, student in Dnipro University of Technology and exchange student in Reutlingen University,

Overview

Agile is a project management philosophy that employs a set of principles and values to help software teams respond to change. Agile teams value individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following a plan. These values were set down in the Agile Manifesto along with 12 principles behind the manifesto [1].

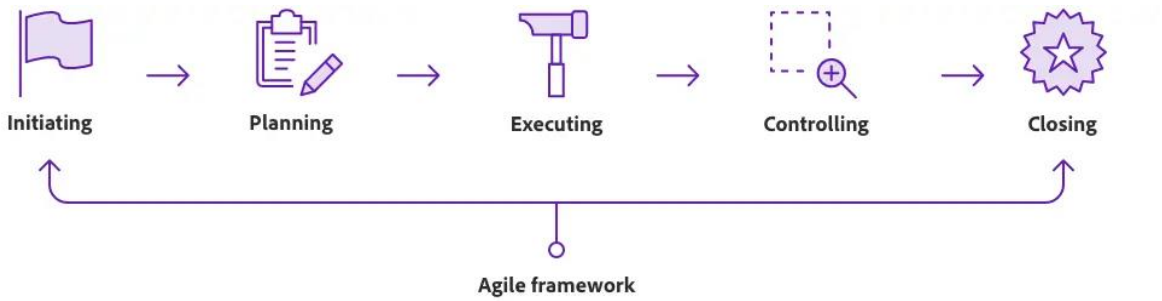
An agile framework is a set of rules, processes, roles and practices that enable agile working in project management and product development. It provides a clear structure and approach to using agile methods in an organized manner.

Agile frameworks are based on the four agile values: customer centricity, adaptability, impact and connection. With these as a foundation, they define a

framework to organize the collaboration of a group. In order for a team to be successful with them, it not only needs the frameworks, but also the appropriate, agile attitude [2].

Five phases of project management to optimize with Agile frameworks

The Project Management Institute maps out five phases of project management to optimize, no matter what framework you use to guide your work.



1. Initiating — kicking off the project with all the requirements.
2. Planning — crafting a strategy to break the project into manageable parts.
3. Executing — carrying through with the work on a task-by-task basis.
4. Controlling — reviewing progress and testing for accuracy along the way.
5. Closing — completing the entire project according to the provided specifications.

Understanding these phases can help to choose the right framework for unique needs. While Agile frameworks tend to be less rigid when it comes to step-by-step processes and deliverables, their strength lies in their ability to define responsibilities and roles while increasing the flow of communication between project contributors [3].

Nexus

Overview

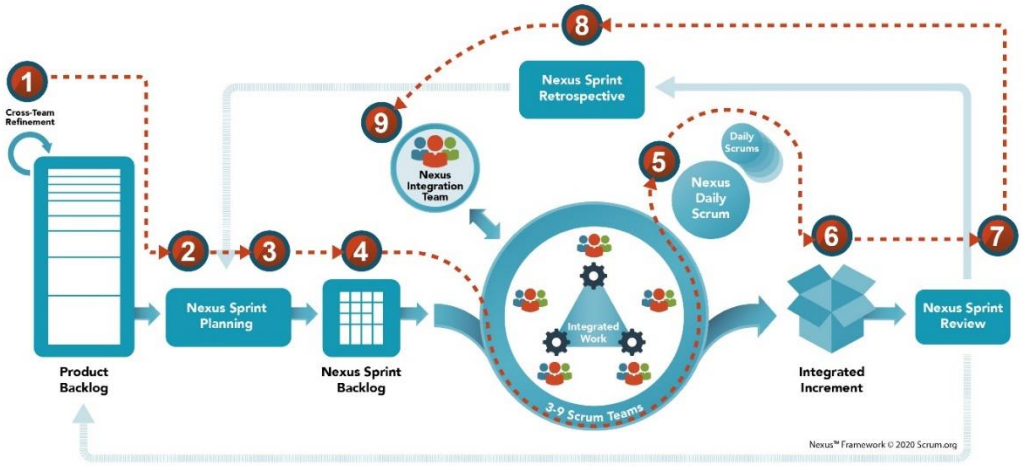
A Nexus is a group of approximately three to nine Scrum Teams that work together to deliver a single product; it is a connection between people and things. A Nexus has a single Product Owner who manages a single Product Backlog from which the Scrum Teams work.

The Nexus framework defines the accountabilities, events, and artifacts that bind and weave together the work of the Scrum Teams in a Nexus. Nexus builds upon Scrum’s foundation, and its parts will be familiar to those who have used Scrum. It minimally extends the Scrum framework only where absolutely necessary to enable multiple teams to work from a single Product Backlog to build an Integrated Increment that meets a goal.

Nexus builds upon Scrum by enhancing the foundational elements of Scrum in ways that help solve the dependency and collaboration challenges of cross-team work. Nexus (see Figure 1) reveals an empirical process that closely mirrors Scrum.

Nexus extends Scrum in the following ways:

1. Accountabilities: The Nexus Integration Team ensures that the Nexus delivers a valuable, useful Integrated Increment at least once every Sprint. The Nexus Integration Team consists of the Product Owner, a Scrum Master, and Nexus Integration Team Members.
2. Events: Events are appended to, placed around, or replace regular Scrum events to augment them. As modified, they serve both the overall effort of all Scrum Teams in the Nexus, and each individual team. A Nexus Sprint Goal is the objective for the Sprint.
3. Artifacts: All Scrum Teams use the same, single Product Backlog. As the Product Backlog items are refined and made ready, indicators of which team will most likely do the work inside a Sprint are made transparent. A Nexus Sprint Backlog exists to assist with transparency during the Sprint. The Integrated Increment represents the current sum of all integrated work completed by a Nexus.



Accountabilities in Nexus

A Nexus consists of Scrum Teams that work together toward a Product Goal. The Scrum framework defines three specific sets of accountabilities within a Scrum Team: the Developers, the Product Owner, and the Scrum Master. These accountabilities are prescribed in the Scrum Guide. In Nexus, an additional accountability is introduced, the Nexus Integration Team.

Nexus Integration Team

The Nexus Integration Team is accountable for ensuring that a done Integrated Increment (the combined work completed by a Nexus) is produced at least once a Sprint. It provides the focus that makes possible the accountability of

multiple Scrum Teams to come together to create valuable, useful Increments, as prescribed in Scrum.

The Nexus Integration Team consists of:

1. The Product Owner: A Nexus works off a single Product Backlog, and as described in Scrum, a Product Backlog has a single Product Owner who has the final say on its contents. The Product Owner is accountable for maximizing the value of the product and the work performed and integrated by the Scrum Teams in a Nexus. The Product Owner is also accountable for effective Product Backlog management. How this is done may vary widely across organizations, Nexuses, Scrum Teams, and individuals.
2. A Scrum Master: The Scrum Master in the Nexus Integration Team is accountable for ensuring the Nexus framework is understood and enacted as described in the Nexus Guide. This Scrum Master may also be a Scrum Master in one or more of the Scrum Teams in the Nexus.
3. One or more Nexus Integration Team Members: The Nexus Integration Team often consists of Scrum Team members who help the Scrum Teams to adopt tools and practices that contribute to the Scrum Teams' ability to deliver a valuable and useful Integrated Increment that frequently meets the Definition of Done.

Nexus Events

Nexus adds to or extends the events defined by Scrum. The duration of Nexus events is guided by the length of the corresponding events in the Scrum Guide. They are timeboxed in addition to their corresponding Scrum events.

At scale, it may not be practical for all members of the Nexus to participate to share information or to come to an agreement. Except where noted, Nexus events are attended by whichever members of the Nexus are needed to achieve the intended outcome of the event most effectively. Nexus events consist of:

1. The Sprint. A Sprint in Nexus is the same as in Scrum.
2. Cross-Team Refinement
3. Nexus Sprint Planning The purpose of Nexus Sprint Planning is to coordinate the activities of all Scrum Teams within a Nexus for a single Sprint. Appropriate representatives from each Scrum Team and the Product Owner meet to plan the Sprint.
4. Nexus Daily Scrum
5. Nexus Sprint Review
6. Nexus Sprint Retrospective

Nexus Artifacts and Commitments

Artifacts represent work or value, and are designed to maximize transparency, as described in the Scrum Guide. The Nexus Integration Team works with the Scrum Teams within a Nexus to ensure that transparency is achieved across all artifacts and that the state of the Integrated Increment is

widely understood. Product Backlog, Nexus Sprint Backlog, Integrated Increment [4].

Sources

1 https://www.atlassian.com/de/agile/agile-at-scale/what-is-safe
2 https://www.me-company.de/magazin/agile-framework/
3 https://business.adobe.com/blog/basics/agile-frameworks
4 https://www.scrum.org/resources/nexus-guide

UDC 004.4

System Analysis in Software Engineering: Applying the Algorithm for Enumerating All Possible Scenarios

Anton Mormul, student, Mormul.A.S@nmu.one, Dnipro University of Technology

Larysa Koriashkina, Candidate of Sciences, Assoc. Prof., koriashkina.l.s@nmu.one, Dnipro University of Technology

Svitlana Kostrytska, Head of the Department of Foreign Languages, Prof., kostrytska.s.i@nmu.one, Dnipro University of Technology

System analysis is the process of studying a system and its requirements in order to design an information system that meets those requirements. It is a key step in the software development performed after project planning. The goal of system analysis is to provide a detailed and complete understanding of the system functionality that needs to be developed [4].

One of the challenges in system analysis is identifying all possible scenarios and occasions that need to be considered when designing the system architecture and writing the technical specifications [1]. For complex systems with multiple variable parameters, it can be difficult for analysts to methodically think through all permutations of events, user actions, and system responses. Critical factors may be missed, resulting in gaps in requirements and unexpected system behavior. The application of a structured algorithm for enumerating all possible scenarios during system analysis is demonstrated in Figure 1. The technique involves defining all parameters affecting the system functionality, calculating the total number of scenarios using combinatorics, and iteratively stepping through the scenarios.