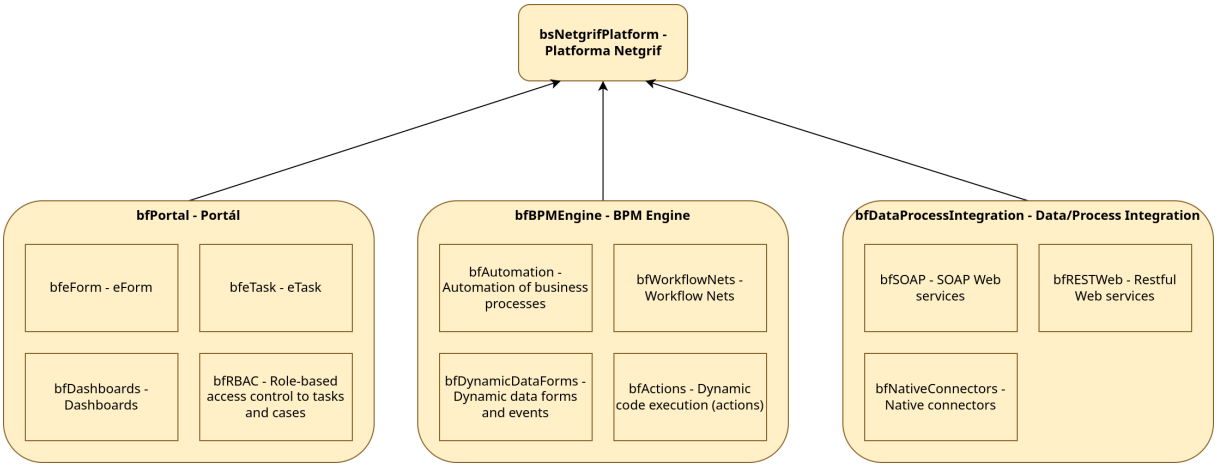
Architecture of the Netgrif Application Platform

1 Business architecture

Netgrif (bsNetgrifPlatfrom) is a low code application platform that allows you to cover end-to-end processes. The platform provides a fast, simple, and efficient way to automate, orchestrate and integrate business processes. It can be used when it comes to the development of new applications or when it is necessary to upgrade older systems. The Netgrif platform is built on the latest open-source and source-available technologies such as MongoDB, Elasticsearch, Spring Boot and Angular. It is easily deployed in small infrastructure. The Netgrif application platform helps those who need to quickly digitize existing or new business processes.

Basic attributes of the Netgrif platform:

- Portal (bfPortal) - one place where tasks are performed by employees (eTasks), but also by customers (eForms).
- BPM Engine (bfBPMEngine) - automated business processes that function as a sequence of tasks that can be run based on business rules. Automation is performed using no code / low code - pro code digitization of "paper jobs"
- Data / Process Integration (bfDataProcessIntegration) - integration of tasks / data from existing applications.
    - Web services implemented using the REST API.
    - Web services implemented using SOAP.
    - Integration using native connector to a target system (ie. system SDK)



Picture 1 – Business architecture of Netgrif Application Platform

Business features and functions of applications created using the Netgrif platform:

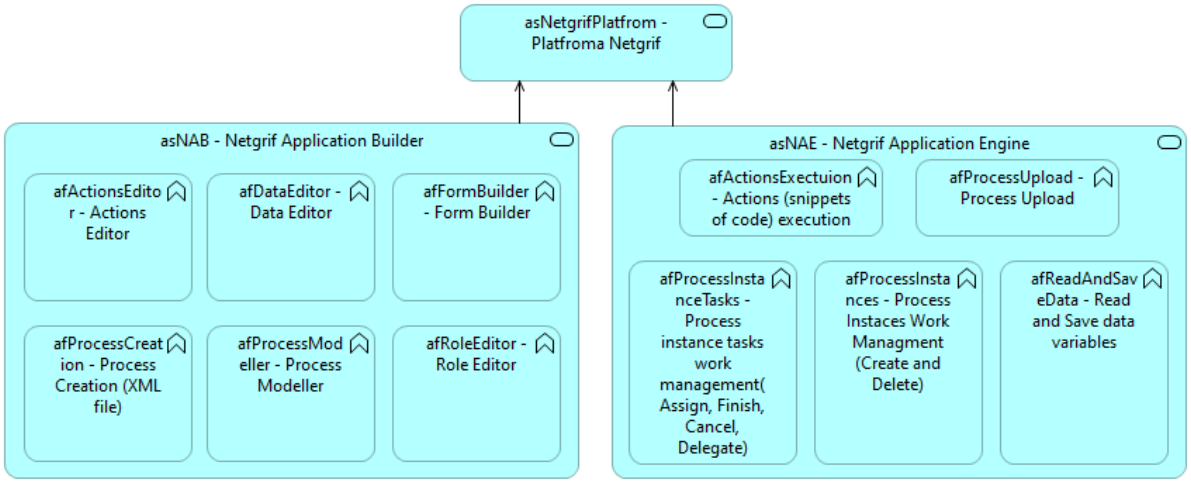
- eForm, eTask (bfeForm, bfeTask) - NAE-based applications provide a single access point to perform employee tasks across the organization. If necessary, also within the business processes performed by customers within the provided services.
- Automation (bfAutomation) - The Netgrif platform allows you to model and automate business processes as sequences of tasks performed in accordance with defined rules, integrate tasks from existing applications into processes, and digitize "paper" tasks with the need to program only to the necessary extent.
- Workflow nets (bfWorkflowNets) - Workflow processes are implemented within Netgrif AE using the so-called workflow nets that consist of state variables, tasks, and their

connections. Workflow nets determine the state in which individual tasks can be performed.

- Dashboards (bfDashboards) - Reports of performed tasks can be displayed in various forms according to customer requirements, for example in the form of graphs, charts, counters or tables. After clicking on a dashboard panel/component, the user can be redirected to a list of tasks available on the Task Overview screen.
- RBAC (bfRBAC) - Access management of the platform is based on the principle of roles and groups. You can define permissions for a role on tasks. But also, you can assign roles to users who interact with tasks in the process. In addition, it is possible to fully automate tasks with the use of the so-called system roles that automatically run tasks. It means that activities are performed automatically by some software instead of a user, e.g., based on a timer or event.
    - Individual tasks in the Task Overview can be assigned to users based on predefined roles. When it is assigned, the job status changes from job processing status to job assignment status. If necessary, the assigned task can be cancelled or moved to another user with the same permissions (reassign).
    - You can use filters to create personalized task reports in applications. Reports can be created by the user or by a supervisor. When managing users, it is possible to assign specific users to roles and groups within the customer's organizational structure.
- Dynamic data forms (bfDynamicDataForms) - Data fields or data forms can be bound to tasks. When performing the task itself, it is possible to fill in data fields or automatically calculate by calling the so-called actions.
    - One data field can be linked to several tasks (via data references) in which its value can be changed or displayed. You can use data field attributes to set data field properties for individual tasks. Using these attributes, it is, therefore, possible to set whether the data field is visible in the form for a given task, whether it is editable, required, and so on.
    - With actions, it is possible to automatically calculate the value of a given data field based on the values of other data fields, including the values of data fields obtained from external systems by calling web services.
- Actions (bfActions) - Small blocks of code that are executed during the life cycle of the modelled process. Actions have access to the application engine API and can interact with the processes, cases, and tasks contained in them. These interactions correspond to an event, such as setting new values, creating new cases, or assigning a task. Multiple actions can be associated with these events, so a chain reaction of events and actions can occur. This allows you to create extraordinarily rich applications from simple building blocks.
- Integrations (bfSOAP, bfRESTWeb, bfNativeConnectors) - Applications created by the Netgrif platform can easily integrate with other systems, databases, applications, etc. This property is achieved by the used technological architecture. You can use any connectors or web services.

2 Application architecture

Netgrif Application Platform (asNetgrifPlatfrom) consists of two tools: Netgrif Application Builder (NAB) and Netgrif Application Engine (NAE).



Picture 2 – Application architecture of Netgrif Application Platform

Netgrif Application Builder (asNAB) is a public web application where users can instantly design their own process models (along with data, data forms, and roles) or import models from any BPMN model.

- Process creation (afProcessCreation) - a tool for creating and editing existing but also new processes and defining their metadata
- Process modeller (afProcessModeller) - a tool for modelling process models in the Petri net format
- Actions Editor (afActionsEditor) - editor for creating and editing actions (pieces of code that are custom and specific for Petriflow language and the Groovy scripting language features)
- Data Editor (afDataEditor) - editor for creating and editing data variables
- Form builder (afFormBuilder) - editor for creating and editing forms related to a specific task
- Role Editor (afRoleEditor) - editor for creating and editing roles belonging to the entire process but also specific tasks

Netgrif Application Engine (asNAE) is a process server engine that executes processes modelled in NAB. It is designed as a Java application with a three-tier architecture that provides a responsive web portal. The web portal generates a dynamic user interface using forms based on imported processes from NAB. NAE is a tool for deploying and running process-driven applications written in Petriflow. It consists of several building elements. The core of NAE is the Process Engine Server, which allows:

- Process Upload (afProcessUpload) - upload, run and delete Petriflow processes;
- Process Instances Work Management (afProcessInstances) - creating and deleting process instances;
- Process instances Tasks Work Management (afProcessInstanceTasks) - assign, finish, delegate and cancel process instance tasks;
- Read and Save data variables (afReadAndSaveData) - work with data variables;

- Actions Execution(afActionsExectuion) - working with actions (functional units of code in the NAE process model that are triggered based on events).

Netgrif application platform supports:

Property	Detail
Working with tasks	The Petriflow language defines tasks as the extension of transitions in the Petri net. When you create a new instance, all tasks are created, and their status is set according to the marking. The user can assign individual active tasks and work with their functionality.
Task delegation	A user working with an assigned task can delegate the task to another user if the RBAC allows it.
Task prioritization	It is possible to assign a priority to individual tasks based on which the user can evaluate the importance of their processing.
Task and process instance filtering	Simple but also advanced filters can be easily created for every task and process instance based on their priority, metadata, data, etc.
Task creation	The purpose of NAB is to create process applications. Petriflow process applications consist of states and tasks. Data, roles, and actions can be easily assigned to all tasks.
Task lifecycle	According to the marking of the process instance, each task can have one of these states: inactive, executable, or running.
Working with data and forms	Data are initialized when a process instance is created. All data are valid for the entire process and can be used in any data form. One form can be created for one task. Tasks and forms can be referenced interprocessly.
Changes in data field attributes	All data have dynamically changeable attributes such as behaviour (visible, editable, required, etc.), style, name, and more. In addition to changeable attributes, dynamic validations can be also used.
Automatic calculations	A programmed piece of code - an action - can be used for individual events in the process. These events involve work with data and tasks.
Responding to events	Petriflow allows you to respond to process instance's events, and task's events and data's events by triggering actions. It is also possible to trigger these events using an action.
Organizational structure management	Organizational structure is solved using a tree structure and is connected to RBAC. It can be mapped to any external IAM system.
Connection to external API	You can use any connectors, web services
Working with files	Files are also one of the data items that can be used in forms. Files can be displayed as a single file, list, or preview of a file.
Logging user activity in processes	All activities of the system and users are logged thanks to events (process instances, tasks and work with data) and thus easily auditable.

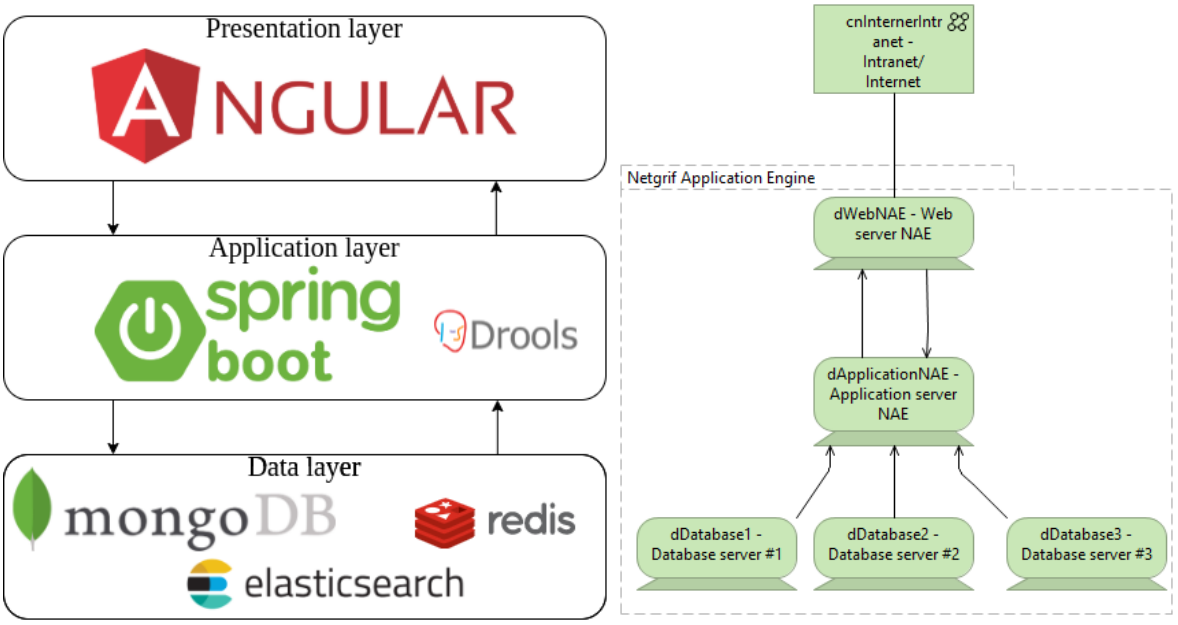
Responsive design	Process applications have optimized content that adapts seamlessly to different screen sizes – PWA.

3 Technological architecture

NAE is a tool for deploying and running process-driven applications written in Petriflow language. It consists of several building elements. The core of NAE is the Process Engine Server, which allows you to:

- load, run, update, and delete Petriflow processes;
- creating and deleting process instances;
- assign, complete, and cancel process instance tasks;
- work with data variables;
- working with Actions (functional units of code in NAE process models that are executed based on events).

NAE is a three-tier web application - it is set up from frontend, backend, and database systems. The frontend in NAE is covered by the Angular platform. The application backend is built on the Spring boot framework. The three databases that make up the data layer are MongoDB, Elasticsearch and Redis. The individual layers communicate with each other using REST API



services. Communication is allowed via HTTP and HTTPS protocols. The standard port for communication with the application backend is 8080. Other standard ports for communication within the solution are MongoDB (27017), Elasticsearch (9200, 9300), Redis (6379). For the correct functioning of the solution, an additional configuration of the application backend and database systems for cluster mode is required, according to the valid manuals.

4 Container architecture

Minimum capacity and performance requirements for NAE technology (specified per container or VM instance). Recommended requirements are 2x minimum requirements.

Container	vCPU	RAM	Storage capacity
Frontend	1	2GB	1GB
Backend	2	4GB	5GB
MongoDB	1	2GB	25GB
Elasticsearch	1	2GB	25GB
Redis	1	2GB	25GB

NAE supports docker images:

- Database server #1 – Elasticsearch – component is implemented by platform „Elasticsearch“. Instancing takes place according to the definition of the container „elasticsearch:7.17.+“.
- Database server #2 – MongoDB – component is implemented by platform „MongoDB“. Instancing takes place according to the definition of the container „mongo:6+“.
- Database server #3 – Redis – component is implemented by platform „Redis“. Instancing takes place according to the definition of the container „redis:6+“.
- Frontend - Web server NAE– component is implemented by platform „Angular“. Instancing takes place according to the definition of the container „NAE Frontend image“, this being a derivative of the general definition of a container „Nginx image“.
- Backend - Application server NAE – component is implemented by platform „Spring boot“. Instancing takes place according to the definition of the container „NAE Backend image“, this being a derivative of the general definition of a container „openjdk:11-jdk image“.

5 Security architecture

The NAE meets these safety requirements (picture 5):

- HTTPS protocol support with TLS 1.2 and 1.3
- securing a solution against OWASP Top 10 (10 most important security risks of web applications)
- communication within components (frontend, backend) is always authorized using authorization tokens (opaque JWT, OpenID Connect, SAML2.0)
- each request is processed based on the set of valid user permissions (roles, permission level in the application, group membership)
- communication between the application backend and databases is secured by an encrypted communication channel and authorization that is not shared with any other entity
- The encryption level of user data can be configured as needed
- application data is encrypted on the application backend side
- The solution supports a secure variant of standard communication protocols such as LDAPS, SMTPS, IMAPS, etc.

6 Data models, flows and standards

The NAE application contains four main data streams and models. They are:

- Communication between backend and frontend. The backend (Spring boot) and frontend (Angular) communicate with each other via the REST API. HTTP and HTTPS requests for working with CRUD operations are used. The RESTful API is extended by the HATEOAS principle. HATEOAS allows you to work with resources published in the API based on hyperlinks that express the allowed operations on the resource at the moment of a request.
- MongoDB is used as the main database. The document data model implemented in this database is an efficient and fast tool for storing and retrieving data. It belongs to NoSQL databases and is an important building block for "real-time" web applications such as NAE. This database stores all data from Petriflow processes, as well as information about users, groups, roles, etc. Communication with the MongoDB database takes place via a proprietary protocol based on the TCP protocol. The connection is secured using the TLS / SSL protocol and authorization.
- Elasticsearch is implemented for quick search in the application. This component contains a distributed full-text search engine. MongoDB has its limit in the form of quick search and indexing. For this reason, NAE applications use Elasticsearch to index data primarily stored in MongoDB. It communicates with Elasticsearch via the database Rest API, the HTTPS protocol.
- The in-memory data structure, Redis, is used as a "cache" and "session store" within the NAE application. Redis also uses a TCP connection with the proprietary protocol called RESP (Redis Serialization Protocol).

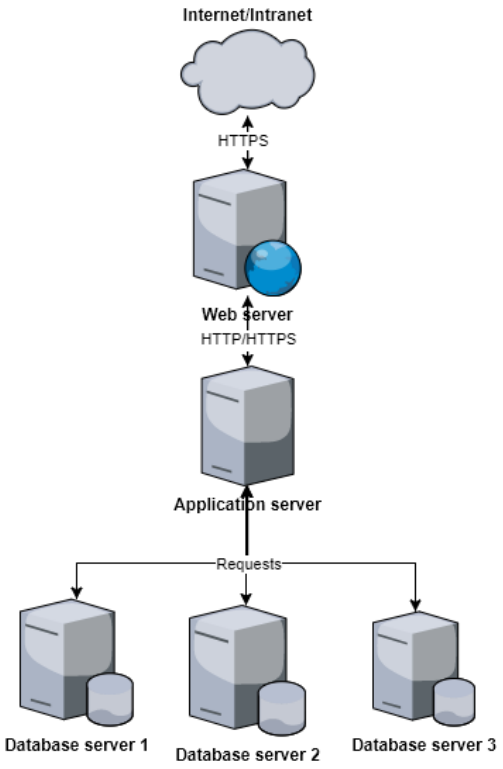


Figure 5 – Data models, flows and standards in NAE

7 NAE Editions

Netgrif Application Engine is available in two editions:

7.1 Community Edition

The community edition of the Application Engine is publicly available, open-source, distributed on the GitHub platform. The edition contains a full-fledged Petriflow workflow engine for building any application. It also includes built-in user management with the ability to connect to LDAP-compliant systems, work with organizational structures, advanced data indexing with Elasticsearch, a full-featured email client, in-process PDF file generator and support for custom business rules across processes using BRE (Business Rules Engine) Drools.

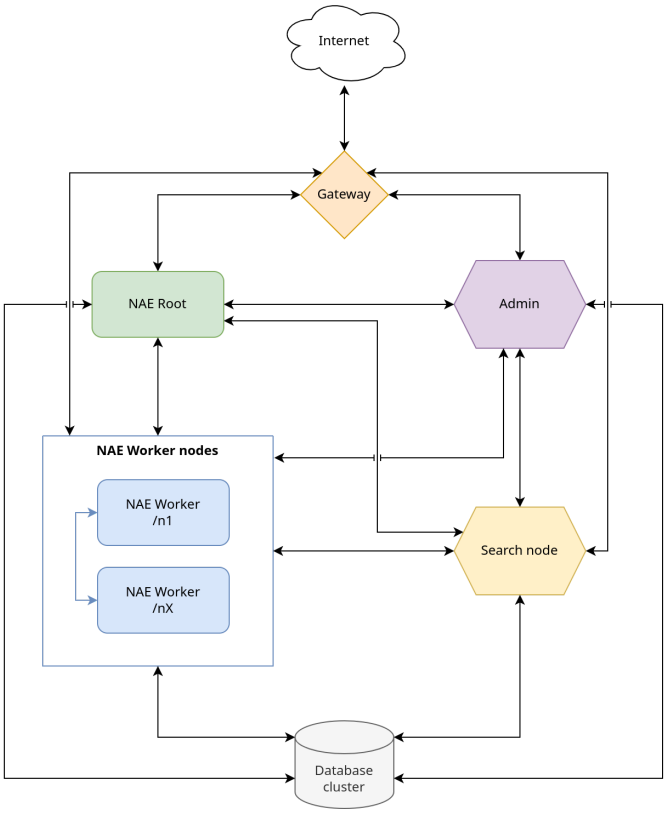
The community edition is available as a standalone Java application, as a library that can be downloaded using the Maven dependency management system, or as a Docker Image for direct deployment to a container environment.

The community edition is intended for development and applications for which one installation of the Application Engine is sufficient. It is suitable for beginning developers, for rapid prototyping of the solution, or for applications that do not require extensive scaling or a larger number of integrations.

7.2 Enterprise Edition

Enterprise edition expands the community edition, so it builds on the same foundations, so a quick and easy migration from the community edition is guaranteed. The Enterprise Edition expands NAE with the ability to deploy to a private cloud environment with support for simple horizontal scaling through direct communication with the Kubernetes Orchestra. It also offers support for authorization protocols OpenID Connect, SAML2.0 with SSO capability and advanced settings for LDAP.

For easier integration with external systems, the edition enables automated generation of connectors for web services with the available OpenAPI v3 or SOAP specifications. It is also possible to publish web services directly from the process without the need for developer intervention.



In addition to NAE, the Enterprise Edition also includes other components to cover the needs of the cloud environment. Netgrif Admin is a component that communicates directly with Kubernetes for automatic scaling and deployment of NAE instances (containers) and for ensuring fast and easy communication between processes no matter where in the cluster the process is deployed. Individual NAE instances communicate using the gRPC protocol. Netgrif Gateway is a component for easier communication of the NAE cluster with the outside world, whether it is communication with the frontend application or external systems. The component automatically redirects queries to the correct process and provides verification of access to deployed processes. The Search node is a component of the cluster for cluster search requests in case when it is not possible to detect a process context of the request so it is not possible to accurately route the request to NAE node. Every cluster deployment has node called NAE Root node which handels system process application (ie. filter management). The last component of the cluster is NAE Worker node which handles all process applications deployed into the cluster. Worker nodes are possible to scale according to system needs.

A significant expansion is the support of plugins as microservices. Plugin is a component of a cluster solution for extending the functionality of the NAE application outside the deployed processes, it is a support function or connector for external services or systems. The plugin is deployed and managed according to the architectural design of the microservices. For comparison with the community edition, the generation of PDF files is considered a plugin and so it can be scaled separately and can be used by all processes deployed in the cluster. Plugin can extend the Action API for process application with new functionality and react to events of NAE nodes and process applications.

7.3 Comparison of the editions

The following table compares the Community and Enterprise versions for a better overview.

	Community	Enterprise
Licence type	Open Source	Proprietary
Workflow Engine	✔	✔
User management	✔	✔
LDAP		
Authentication & Profile	✔	✔
Groups		✔
Mail client	✔	✔
Business Rules Engine Drools	✔	✔
Event Log / Audit Log	✔	✔
Netgrif Admin		✔
Netgrif Gateway		✔
Netgrif Search node		✔
gRPC protocol		✔
Connection with IDM solutions (OpenID Connect, SAML2.0)		✔
Automated integrations (OpenAPI3, SOAP)		✔
Plugin management		
running locally	✔*	✔
as a microservice		✔
Contractual guarantees		✔
SLA support (up to 24/7/365)		✔

*Plugins included in the Community Edition are PDF generator, QR code generator, and ZIP lookup service. Plugin can be active as locally loaded jar file into NAE instance.