Integrated Information System for the Management of Activities in the Organization

Adriana-Meda UDROIU[1,2], **Ionut SANDU**[1,3]*, **Mihail DUMITRACHE**[1,4]

[1] National Institute for Research and Development in Informatics, 8-10 Mareșal Averescu Avenue, Bucharest, 411515, Romania

[2] Politehnica University of Bucharest, 313 Splaiul Independenţei, Bucharest, 060042, Romania
meda.udroiu@rotld.ro

[3] "Lucian Blaga" University of Sibiu, 10 Victoriei Boulevard, Sibiu, 550024, Romania
ionut.sandu@rotld.ro (*Corresponding author*)

[4] University of Bucharest, Faculty of Letters, 5-7 Edgar Quinet Street, Bucharest, 010017, Romania
mihail.dumitrache@rotld.ro

Abstract: This paper presents the implementation of an integrated information system whose purpose is to ensure the management of activities within an organization. Such systems are successfully used in organizations, ensuring the automation of the organizational flow and the efficient and effective management of organizational resources. The architecture of such systems is offered for exploitation by recognized companies, under license. Specific to this system is the approach using open-source software tools, as well as the modularization of the application, which allows the independence of its installation from the existing platforms of the beneficiary. The obtained prototype is the result of a research project, carried out over a period of three years, with direct applicability to the beneficiary and the extension, subsequently, to other organizations. The beneficiaries of the system are public entities, which is why the design, development and implementation require specific conditions determined by the legislative and administrative constraints of the functioning framework of the governmental institutions.

Keywords: Interoperability, Middleware platform, Infrastructure services, Operational security.

1. Introduction

The pandemic period has shown us the importance of the existence of a well-structured system at the level of the organization, which can efficiently and effectively manage all the processes in the organization. From working from home to making remote meetings an information system is needed to correctly manage all the resources of the organization.

Integrated information systems for the management of activities ensure the management of information, resources and activities in the organization, to determine optimal workflows for the management of tasks at the level of each structure.

It is known that all information platforms within an organization, regardless of their functional spectrum, shape the concepts of *information resource* type, *business process* and *identity.* It is also known that business processes cross the border of a single software platform, and all the platforms must have a unified reference on the resources, processes and identities involved in the business processes in the organization. Considering the above two hypotheses, there is a need for a platform, at the level of the information system that ensures coherence and logical cohesion of all the existing platforms and provides enrollment capacities in the information system, through clear methods, of any other systems that come to complete the functional spectrum that the organization needs (Han & Sun, 2016; Mohamed et al., 2013). This type of platform is found in the specialized literature under the name of "*middleware platform*" and it has the role of "standing between" the software components to ensure the communication and mediation process.

Specifically, this "middleware platform" represents the backbone of the information system and provides the organization with the so-called "*infrastructure services*" that serve, in a unitary way, the component entities of the informational system (Jones et al., 2017).

In the design process of the platform the following objectives will be approached: functional decoupling through SOA architecture, standardization of the way of interaction between components, the modeling principles standardization, creating a reusable infrastructure service layer at organizational level, creating the single centralized identity service and generating the premises for compliance with the GDPR requests regarding the processing of personal

https://doi.org/10.24846/v31i2y202203

data, expandability of the platform by creating connectors, securing the system at all its levels (Mohamed et al., 2013; Zhang et al., 2017).

The work focuses on the implementation of the system and on the its functionality requirements depending on the specificities of the beneficiary's platforms. In this regard, Section 2 briefly describes the structure of the system, the architecture and the model that was the basis for the elaboration of the system, with the enumeration of the open-source tools and platforms used.

Section 3 presents the ways and solutions used to implement the system, a special contribution being put on data security solutions. They have been designed in such a way that they can ensure the confidentiality, integrity and availability of data in parallel with a rigorous access control.

Section 4 describes the test methods of the system, customizing the requirements of the beneficiary and populating and running with data and test information to simulate as well as possible the real operating conditions.

The last section outlines the conclusions of the present work and summarizes the main elements obtained after testing and their adaptation to the real operating conditions.

2. IISMA Architecture and Conceptual Model

IISMA (Integrated Information System for Management of Activities) ensures the fluidization of the information flow, the improvement of the resource management, the inter-institutional interoperability by offering a versatile, safe and resilient solution that can be customized according to the specific requirements of the public entities (Mohamed et al., 2013).

IISMA is based on open-source solutions and functional specialization of software modules, aiming at conceptual separation to reduce complexity. IISMA ensures: functional decoupling of modules (with preservation of interoperability), standardization of functional modules, implementation of extensibility facilities, reusing facilities for components and securing the processed and stored data. Finally, IISMA will be integrated and tested at the level of public entities (Mohamed et al., 2013; Stock, Greis & Kasarda, 1998).

The design, development, implementation and testing of the system consider the major objectives it had to meet, namely:

- Automation of the management of processes and activities of public institutions – considering the rapid and easy development of working activities with the development, monitoring, updating and real-time optimization of the existing procedures;

- Automation of the management of the programs / projects of the institution – by automating the development, management and monitoring of programs and projects;

- Automation of resource management, which includes dynamic management of asset and inventory record-keeping, handling and reporting activities, administrative and material, financial and human resources management activities;

- Automated management of the internal information circuit, of documents in electronic format and of the transfer of documents between institutions, fulfilling the technical and procedural requirements of information security and ensuring a unitary framework for the management of information and reports that offers the possibility of establishing the common agenda between specific entities. To achieve a viable solution for national security, the present approach also aims to develop the public-public and public-private partnership, by creating an innovative product which concentrates specific knowledge and human and material resources from public institutions and the relevant national industry (Stephens, 2007; Praeg & Spath, 2010).

The functional architecture and operational modules of the system are only mentioned, because they were extensively presented in the paper "Implementing an Integrated Information System designed for Romanian Public Entities", published in the *SIC* journal, in 2018. The component modules were presented to show how the implementation and testing of the system were carried out at the beneficiary. Because the constituent elements of the modules and the interdependence between them will be analyzed, it was considered useful to remember them:

M1. Integrated identity and access rights management module. Unique authentication system (IISMA-ID code). It ensures the unitary administration of user accounts, authorization rules and provides, at the platform level, multi-factor secure authentication mechanisms, with the following sub-modules: Multifactor authentication, Multiple user source administration, "Service Provider" systems administration, "Workflows" configuration of approval user, adjustment actions and authorizations, Configuration of notification message templates.

M2. The process, procedures and activity management module provides functional capabilities for modeling and controlling the logical structure of processes, procedures and activities, with the following sub-modules: Dashboard, Processes, procedures and activities, Reports and performance indicators, Monitoring processes, procedures and activities, Managing process definitions, procedures and activities, Modelling of process and procedure definitions.

M3. Management of the programs and projects of the institution. Depending on the institutional development strategy, the institution may prepare, prior to the identification of funding sources, a set of project proposals, by framework objectives, evaluated and classified internally, at the level of the structure of the strategy, with the following sub-modules: Programs Administration, Project Registration, Project Evaluation and Approval, Ongoing Projects, Control and Monitoring, Reports and Performance Indicators.

M4. Management of institution resources with sub-modules: Modeling of resource types and classifiers, Material and financial planning, Treasury, Supply marketing and procurement, Stocks, Production, Logistics, Human Resources.

M5. Document and communication management with Modeling document types and classification schemes, Modeling number registers, Modeling approval processes, Recording document entries and outputs.

M6. Maps (GIS) with Map Management and GIS Data Analysis.

M7. Analytics and dashboard (BI) with sub-modules: Configuration of data sources and report models, Scheduling of periodic reporting run, Reports "AD HOC".

M8. Notifications with Active Notifications View, Submit Notifications.

M9. Technical monitoring platform parameters with Monitoring processor load, Memory, Threads, Storage space, Platform usage statistics monitoring, System logs monitoring.

M10. Platform availability monitoring with configuration monitoring rules, configuration notification rules (e-mail, SMS), availability monitoring.

In addition to the 10 operational modules, it is considered necessary to have a module **for monitoring the security of the** IISMA **communication network.** This module contains the following functional blocks: Network traffic analysis component (Syslog-NG, Bro), Network intrusion detection component (Suricata), Data collection and processing / filtering component (Logstash), Centralized data storage, indexing and analysis component (Elasticsearch), Alerts visualization and data consultation component (Kibana).

The entire integrated information system (IIS), which contains the 10 modules mentioned above, as well as the related security systems are modulated through open-source applications.

Also, the main tools and software platforms and their roles used for the implementation of the system are the following:

- WSO2 Identity Server: identity management and unitary authorization at platform level (WSO2 Identity Server, n.d.);
- OPEN LDAP: storage credentials of platform users;
- WSO2 Enterprise Integrator with modules: WSO2 Enterprise Service Bus (WSO2 Enterprise Service Bus, n.d.) role: publishing platform-wide integration services; WSO2 Business Process Server (WSO2 Business Process Server, n.d.), role: support for BPMN and WS-BPEL business processes at the platform level; WSO2 Message Broker (WSO2 Message Broke, n.d.) role: technical messaging via MQTT at the platform level; WSO2 Analytics, generic analysis engine role (WSO2 Enterprise Integrator, n.d.);

- WSO2 Business Activity Monitor (BAM) analysis and publication of graphic results on business processes (WSO2 Business Activity Monitor, n.d.);

- WSO2 Governance Registry: administering the registry of information resources and configuring taxonomies (WSO2 Governance Registry, n.d.);

- WSO2 Application Server: application server (WSO2 Application Server, n.d.);

- WSO2 Developer Studio: development of WSO2 platform components, graphic modeling of business processes (WSO2 Developer Studio, n.d.);

- Report Server Community: BI engine, construction of ad-hoc reports, graphic, “drag and drop”, over multiple data sources or their programming (Report Server Community, n.d.);

- NGINX: load balancer for application servers and infrastructure services (NGINX, n.d.);

- PostgreSQL: relational database (Postgres SGBD);

- HA Proxy: Database cluster connections manager (HA Proxy, n.d.);

- Patrons: database configuration management for promoting a slave node of the cluster in the master node in case of master crash;

- ZABBIX: technical monitoring and availability of the platform (ZABBIX, n.d.). All system modules can run on Linux platform (operating system). All libraries used to make the platform are available with “open-source” license.

3. Implementation of the System

The implementation of the conceptual elements is processed with the GitLab platform (GitLab, n.d.) for the entire work process, starting with maintaining the versions of the source code of the system and up to the installation of binary files, the result of compilation, in the centralized image register of the IIS.

Regarding the way of encapsulation of the binary images of the functional modules, the DOCKER (DOCKER, n.d.) is used. It offers very good premises for an easy and secure installation, respectively the running of the functional modules of the IIS, respectively the possibility of installing the IIS modules in any other location by taking them over from the image register. This option is very important because the IISMA must be installed in multiple locations.

In the process of conceptualizing the data model, it was considered that, no matter how well it was defined in an initial analysis step, it may subsequently have to adapt or expand.

Specifically, the SGBD used being PostgreSQL, it has the unstructured JSON (JSON, n.d.) and JSONB data types alongside the operators and functions that allow working with these types of data (Postgres SGBD, n.d.).

Regarding JSON data structures, they are particularly used nowadays, being an alternative to XML structures.

The conceptualization of administration module of the data structures from the IIS modules, is achieved by using a graphical editing tool with the capacity of modeling by using DRAG & DROP method which automatically generates the structure of the JSON data model. This editor will be available in all IIS modules for creating and editing the structure of nomenclatures, respectively for creating and editing data forms.

The conceptualization of the system security is achieved through the following elements:

- Perimeter security elements – securing access to the communication network of the IIS;

- Elements of internal security of the communication network – securing data transmissions within the communication network of the IIS;

- Operational security elements – securing access to data within the IIS modules.

At the level of the IIS modules, an important aspect is that of controlling and monitoring the access to data by authorized and unauthorized users. In this respect, the conceptual model of data security considers (Figure 1):

- The users’ belonging to the organizational structure modeled through the Human resources module;

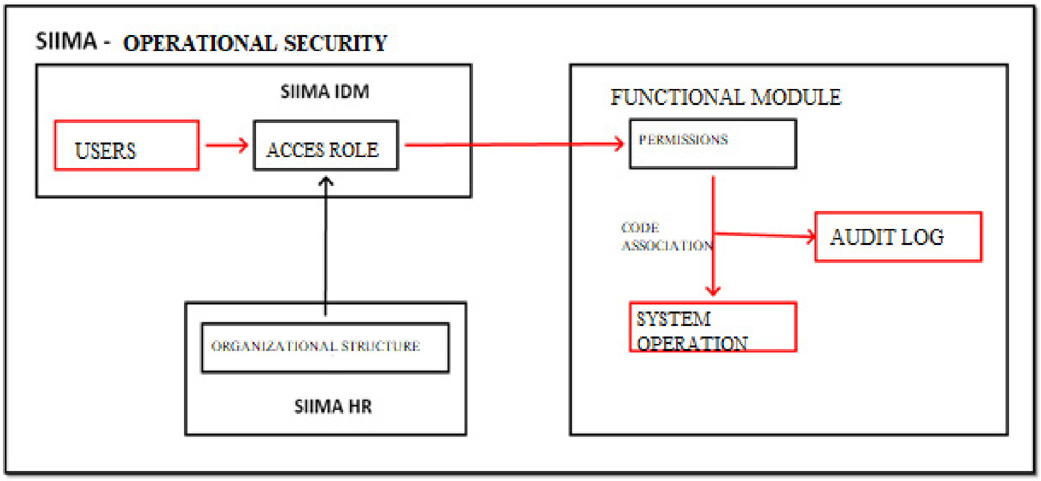

Figure 1. Concept of internal, operational security

- The dependency of the roles on the nodes of the organizational structure modeled in the Human resources module;

- The dependency of the types of access permissions on the specificity of the functional blocks available at the level of each functional module, each one separately.

For network security Security Onion is used which is a free, open-source Linux distribution dedicated to intrusion detection, security monitoring and log management.

The Security Onion solution is built based on a distributed client-server model. Architecturally, for complex and big data infrastructures, Security Onion recommends multiple data collection and storage nodes.

Within IISMA, the installation and configuration of Security Onion is carried out in stand-alone mode (Figure 2). All the components are located on a single virtual machine, with the possibility for a distributed installation to be performed, depending on the level of traffic to be analyzed.

Suricata is used for intrusion detection, a mature, efficient and robust open-source product that will inspect traffic in the IISMA network, by using rules and signatures, with support for advanced scripting (based on THE LUA language) for the detection of complex threats.

An excellent interface for querying and viewing alerts in Suricata is Squert. It is available via web access, accessible from the Kibana component and it can be used for free within the Security Onion solution.

The data collection, indexing and filtering component uses Logstash, an open-source product from Elastic that is integrated into Security Onion. Kibana is used for viewing alerts and consulting data from Elasticsearch.

The data storage, indexing and analysis component uses Elasticsearch, an open-source product that has the role of storing and indexing the data received from Logstash and allows performing and combining several types of searches - structured, unstructured, geo - in any way through the Kibana interface.

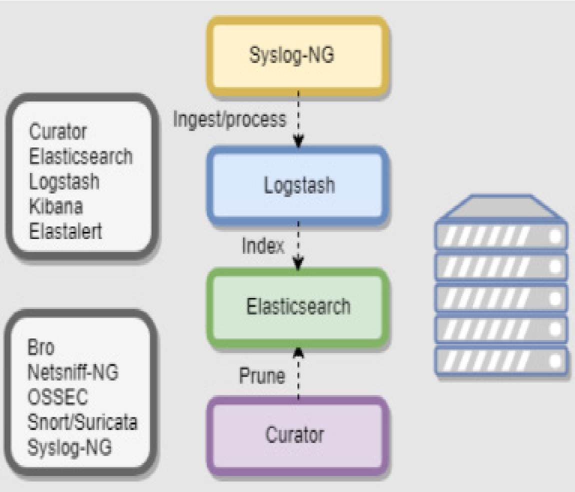

Figure 2. Security Onion stand-alone architecture realized within IISMA (InfoSecAddicts, n.d.)

4. Deployment Model

The logical model of the IISMA system testing infrastructure (Figure 3) is based on running all system components on the Docker infrastructure, namely each component of the system will run within a Docker container. In addition to the containers which are specific to the functional modules of the IISMA system, a container related to the source code management software system, with automated deployment (continuous integration) and quality control (ensured by the Gitlab platform), respectively, was installed.

To run the logical deployment model, it was necessary to have a series of specific configurations in the beneficiary's infrastructure (hardware and software) that would allow the access and installation of the test data. All the restrictions that were requested were considered and all the IISMA modules were implemented, which were subsequently loaded with test data. In the case of ICI servers, these specific configurations were not required. It is worth mentioning that the system works and it is implemented on restrictive platforms in terms of rules of access to information.

The general architecture of the solution execution environment (Figure 4) and the method of installation (Figure 5) are shown in the diagrams below.

The proposed architecture for the installation of IISMA modules is based on Docker containerization, a modern open-source solution internationally recognized for its robustness and reliability, with good technical support and a very large number of implementations in various fields. The Docker-ization process is represented by the installation of an engine (daemon) on a physical

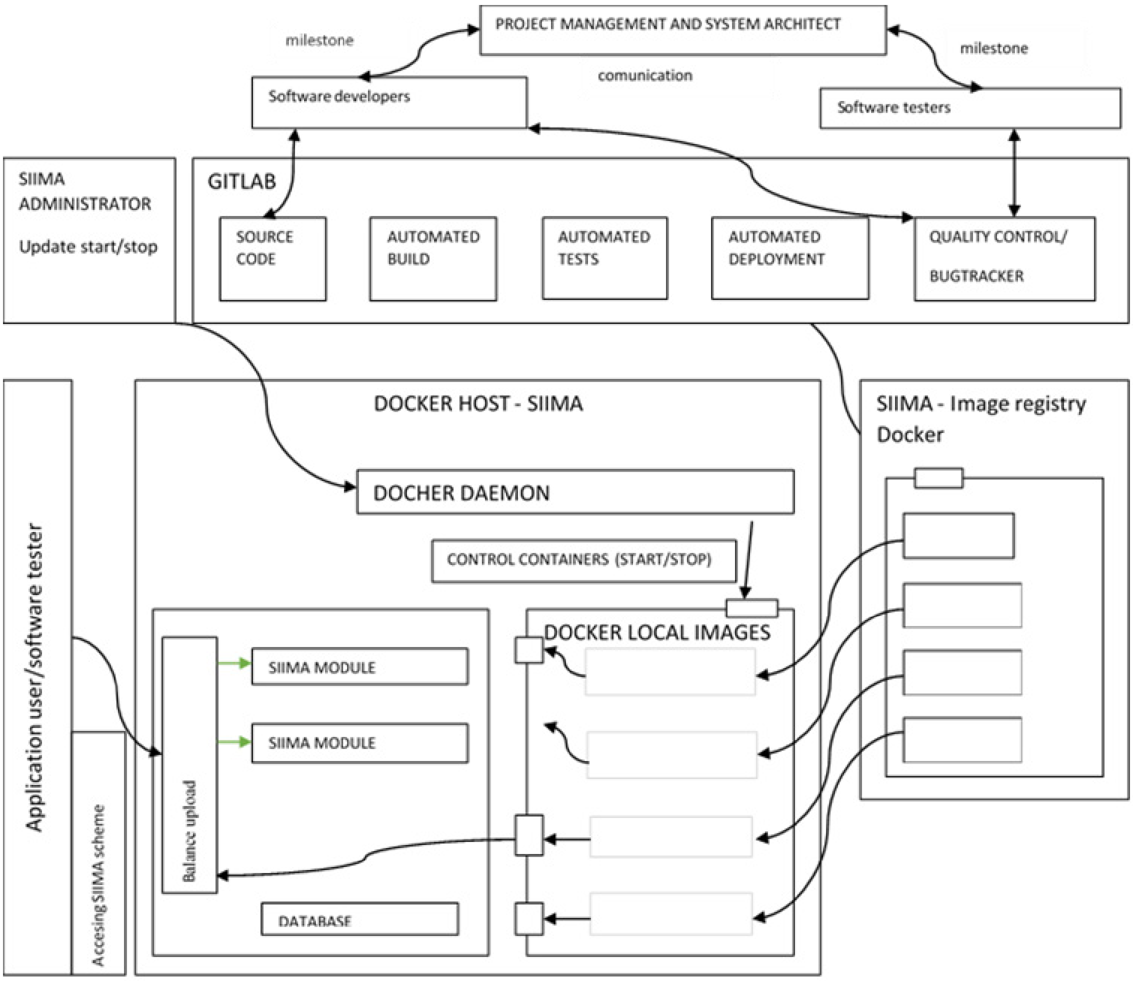

Figure 3. Logical model for the arrangement of the test infrastructure

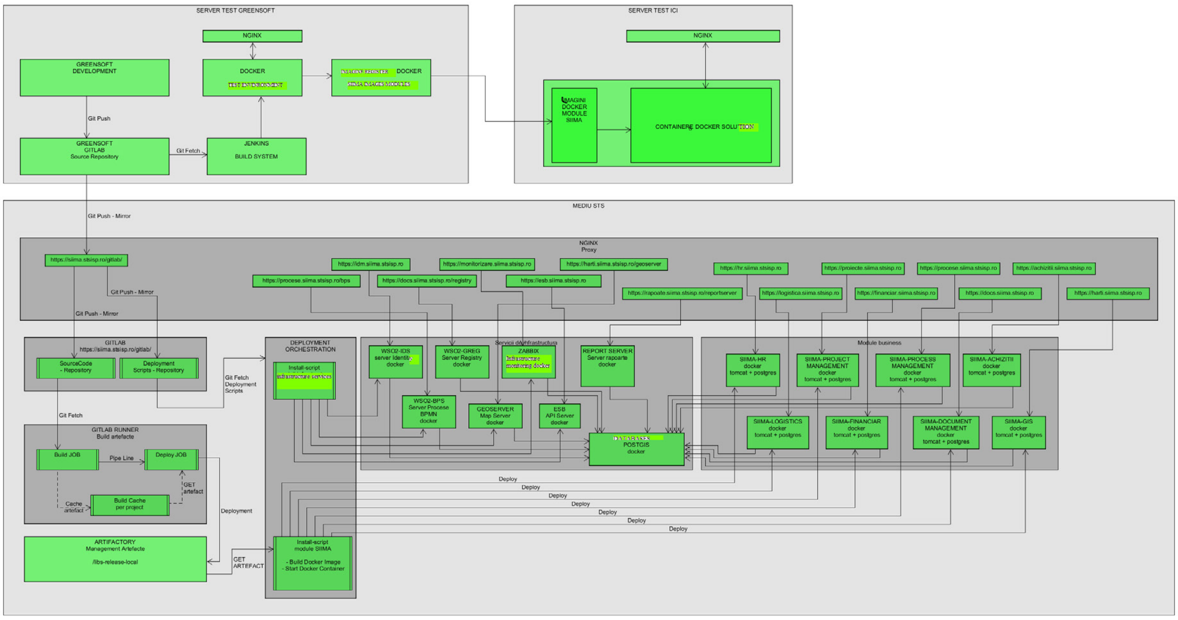
SERVER TEST GREENSOFT
NGINX
GREENSOFT DEVELOPMENT
DOCKER
TEST ENVIRONMENT
IMAGINE REGISTER DOCKER
SIIMA IMAGES MODULES
Git Push
GREENSOFT GITLAB Source Repository
Git Fetch
JENKINS BUILD SYSTEM
SERVER TEST ICI
NGINX
IMAGINI DOCKER MODULE SIIMA
CONTAINERE DOCKER SOLUTION
MEDIU STS
Git Push - Mirror
NGINX Proxy
https://siima.stsisp.ro/gitlab/
https://idm.siima.stsisp.ro
https://monitorizare.siima.stsisp.ro
https://harti.siima.stsisp.ro/geoserver
https://procese.siima.stsisp.ro/bps
https://docs.siima.stsisp.ro/registry
https://esb.siima.stsisp.ro
https://rapoate.siima.stsisp.ro/reportserver
https://hr.siima.stsisp.ro
https://proiecte.siima.stsisp.ro
https://procese.siima.stsisp.ro
https://achizitii.siima.stsisp.ro
https://logistica.siima.stsisp.ro
https://financiar.siima.stsisp.ro
https://docs.siima.stsisp.ro
https://harti.siima.stsisp.ro
Git Push - Mirror
Git Push - Mirror
GITLAB https://siima.stsisp.ro/gitlab/
SourceCode - Repository
Deployment Scripts - Repository
Git Fetch Deployment Scripts
Git Fetch
GITLAB RUNNER Build artefacte
Build JOB
Pipe Line
Deploy JOB
GET artefact
Cache artefact
Build Cache per project
Deployment
ARTIFACTORY Management Artefacte /libs-release-local
GET ARTEFACT
DEPLOYMENT ORCHESTRATION
Install-script infrastructure services
Install-script module SIIMA - Build Docker Image - Start Docker Container
Servicii de infrastructura
WSO2-IDS server Identity docker
WSO2-GREG Server Registry docker
ZABBIX Infrastructure monitoring docker
REPORT SERVER Server rapoarte docker
WSO2-BPS Server Process BPMN docker
GEOSERVER Map Server docker
ESB API Server docker
DATABASES POSTGIS docker
Module business
SIIMA-HR docker tomcat + postgres
SIIMA-PROJECT MANAGEMENT docker tomcat + postgres
SIIMA-PROCESS MANAGEMENT docker tomcat + postgres
SIIMA-ACHIZITII docker tomcat + postgres
SIIMA-LOGISTICS docker tomcat + postgres
SIIMA-FINANCIAR docker tomcat + postgres
SIIMA-DOCUMENT MANAGEMENT docker tomcat + postgres
SIIMA-GIS docker tomcat + postgres
Deploy
Deploy
Deploy
Deploy
Deploy
Deploy
Deploy

Figure 4. General architecture of the execution environment

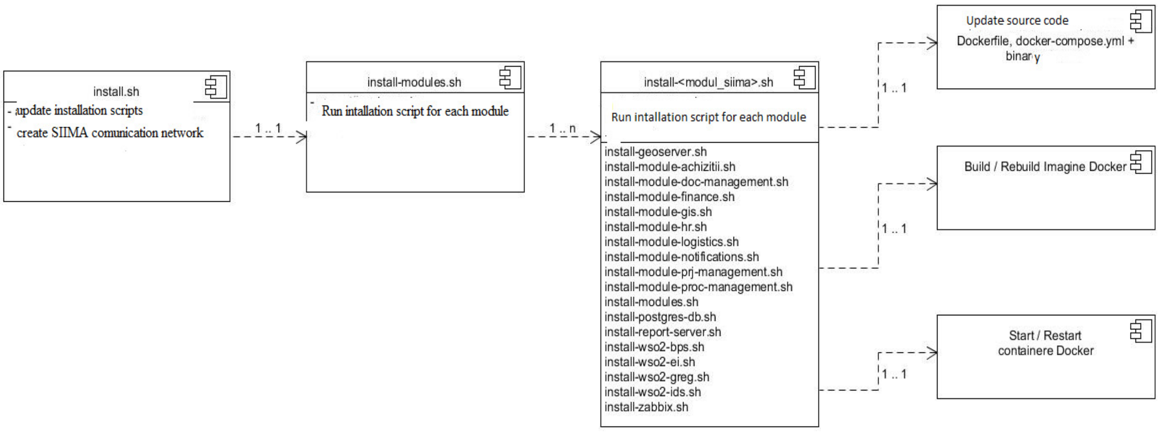
install.sh
- update installation scripts
- create SIIMA comunication network
1 .. 1
install-modules.sh
- Run intallation script for each module
1 .. n
install-<modul_siima>.sh
Run intallation script for each module
install-geoserver.sh
install-module-achizitii.sh
install-module-doc-management.sh
install-module-finance.sh
install-module-gis.sh
install-module-hr.sh
install-module-logistics.sh
install-module-notifications.sh
install-module-prj-management.sh
install-module-proc-management.sh
install-modules.sh
install-postgres-db.sh
install-report-server.sh
install-wso2-bps.sh
install-wso2-ei.sh
install-wso2-greg.sh
install-wso2-ids.sh
install-zabbix.sh
1 .. 1
Update source code
Dockerfile, docker-compose.yml + binary
1 .. 1
Build / Rebuild Imagine Docker
1 .. 1
Start / Restart containere Docker

Figure 5. Method of installation

server, which can transform images (packages optimized for a certain product: Apache Tomcat, Nginx, Postgres, etc.) into containers (software units like virtual machines, but with a much reduced and streamlined resource consumption) that will provide the respective micro-services. The images are available in central or local registers and can be customized to specifications using Docker-file or Docker-Compose. Containers shall be managed centrally or independently, locally or remotely.

In the installed configuration, for the modules testing it is necessary to load SIMA modules together with the test data, as they allow easy access, in relation to the real configuration. In this regard, for each module test data that the user can run are loaded as follows:

M1. Identity management. This software module will be loaded with the pre-generated list of roles for each department within the organizational structure. For each node within the organizational chart, two roles will be generated, one for the position of head of department and one for the regular user within that department, respectively.

M2. Process management of procedures and activities. The initial data for this module will consist of the types of standard business process models namely:

Request process – generic approval (this process model will be used for inter-module requests such as: logistic / human resource request for allocation per project, request to initiate the procurement procedure, etc.);

Logic process – identified at department level (e.g.: the process of awarding military degrees).

M3. Management of the programs and projects of the institution

No eligible test structures have been identified so far.

M4. Management of institution resources

M4.1. Human resources management

Organizational structure. It defines a basic organizational structure with which the HR module is initialized. This organizational structure is hierarchical, on several levels. The recommendation is that the departments be coded so that the code of the parent department is found as a prefix in the child department code.

Initial nomenclatures of qualifications and skills. Because each position is associated with a job description, at the level of organizational node, it is necessary to pre-initialize the nomenclatures specific to the constituent elements of the job description (e.g.: competent nomenclature, nomenclature certifications, nomenclature types of courses, nomenclature types of studies). Each of these nomenclatures can be represented in a CSV file, at the level of each record being characteristic the information *code* (text - 50) and *name* (text - 255).

M4.2. Management of logistics resources

Nomenclatures types of resources. In this module it is necessary to present a nomenclature of types of logistic resources. Each of these nomenclatures can be represented in a CSV file, at the level of each record being characteristic the information *code* (text - 50) and *name* (text - 255).

M4.3. Procurement management. Nomenclatures specific to the types of purchases

M4.4. Financial resources management

Model budget structure. For each annual budget, the draft budget will be initialized based on a predefined, standardized template. The pre-loading of such a structure is carried out by chapters, titles, etc. to facilitate the testing of the budget construction process.

M5. Document and communication management. In the generated lines it is necessary to define the types of standard documents and specific attributes.

M6. Map module (GIS)

Nomenclatures types of geospatial resources. In this module it is necessary to present a nomenclature of geospatial resource types (e.g.: buildings, antennas, e.g.). For each resource type, it is necessary to define the specific attribute types. Each of these nomenclatures can be represented in a CSV file, at the level of each record being characteristic the information *code* (text - 50) and *name* (text - 255), *geometry_type* (point, polygon, line).

Geospatial test datasets. For a type of geospatial resource, a GIS data structure is required (e.g.: a set of streets, buildings, points of interest). Alternatively, you can import some data structures taken from OpenStreetMap (GeoServer, n.d.(a); GeoServer, n.d.(b)).

M7. Analysis and reporting module

Report templates. At the level of each IISMA module, a set of reports is defined to be modeled within the analysis and reporting mode. For example:

- for HR (Human resource): list of personnel per department, list of personnel for whom the legal conditions for granting the military degree have been met, list of personnel for whom it is necessary to update the CV;
- for PM (Project management): list of activities without allocated resources, list of projects with delayed activities, list of projects for which procurements have not been made in time;
- for PR (Procurement): procurement plan, situation of procurement;
- for Financial: situation of budget execution in employment, liquidation, authorization, payment terms;
- for Logistics: Stock situation for a specific management;
- for PMS (Process management): the situation of the processes of a certain type carried out during the period, the situation of the processes in delay, the situation of the activities carried out by a user (process activities), the situation of the activities not carried out on time.

M8. Notification module. This module does not require test data.

M9. Technical monitoring module platform parameter. This module does not require test data.

M10. Platform availability monitoring**.** This module does not require test data.

M11. Security Module.

The testing of the system components was carried out in several stages:

1. Development/customization stage. During this stage, the testing team became the main actor by carrying out activities specific to internal testing (discovery of defects, improvements, new characteristics and problems). Internal testing of the components of the software product IISMA implies the documents considered the coverage of the requirements defined in the technical proposal, as well as the preparation of the IT solution for testing for obtaining acceptance and piloting.

At the end of this stage there was a stable, complete, tested version, ready to be delivered to the beneficiary.

2. Component testing. Component testing was executed by the development team to ensure their functionality and full code testing for acceptance in internal testing. The contribution of the testing team in the testing of the components was one of consultancy / support. The responsibility to ensure a quality control over the testing executed by the development team belonged to the development team, IT and implementation team in this phase.

The following areas of the project had the components tested and approved before being handed over to the test team:

- Unitary testing of the developed functionalities;
- Testing for the integration of web components and services;
- Databases for testing, including stored procedures, the schematic of tables and any data migrations necessary for the testing process, including conversions of the databases used if necessary.

The completion criterion of this stage was Full Code, meaning that all functionalities and components were complete and available to the test team in the test environment.

3. Preparation of intermediate versions of the test. At this stage, intermediate versions of the solution were prepared to be handed over to the test team. They contained operational functionalities, called RC - Release Candidates. Before accepting a version for internal testing, the test team made sure that an appropriate white-box test was performed by the development team. Intermediate versions have passed the following checks to become a candidate in internal testing:

- They were validated by the development team coordinator, who certified that they were complete in terms of the functionalities that needed to be included;
- The location where they were submitted was verified and communicated to the test team;
- Release Notes were checked and contained information about the included changes, issues resolved, etc.;
- The appearance of the graphical interface was “frozen”;
- The structure of the database was “frozen”.

The internal testing took place after the intermediate version had met the above criteria – this being considered a full intermediate version. Subsequent versions contained only changes resulting from the resolution of the problems encountered.

4. Internal testing

For each intermediate version, the following activities were carried out:

- Verification of the fulfillment of the specifications within the Technical Project and those included in the analysis and design meetings;
- Registration within the monitoring application of any deviation from the expected results (incidents, improvements).

Several internal test cycles have been completed for each version handed over to the test team. During the repetition of cycles for the discovery and identification of all defects, the test team carried out various types of tests (functional, non-functional, performance, stress, etc.). The results of the internal testing were communicated to the development team in order to remedy all the identified defects through the dedicated Jira application.

During the repeated cycles of internal testing, frequent and regular meetings were held with the development team and the testing and quality assurance team, meetings that were much more numerous than in the other phases of the project. The test team presented the defects found with the objective of triaging them and determining priorities in allocating tasks to solve them.

The internal testing stage was considered completed when the final version of the IT solution was delivered to the testing team, the specific testing activities were carried out.

5. Test stage (External testing). The external testing was done based on the testing plan.

6. Acceptance testing

Upon entering this phase there was a stable, complete, tested version, ready to be delivered to the Beneficiary (version 1.0 of the solution). This version was installed and configured within the beneficiary's environment.

This phase was also attended by the members of the beneficiary's team responsible for validating the IT solution. The purpose of this phase was to verify through the preliminary acceptance tests that the IT solution corresponds from a qualitative point of view to the Beneficiary's requirements, according to the specifications in the task book and the analysis and design report, and to identify potential operational and security problems before entering the piloting and production phase.

At the level of testing for preliminary acceptance by the beneficiary, the Pass/Fail criterion was:

- 100% of the test scenarios have been executed at least once;
- 0% of defects of severity 1, 2, 3 or 4 are open;
- 100% of the discovered defects of severity 1 or 2 were tested in regression;
- 50% of the discovered defects of severity 3 were tested in regression.

The purpose of this step was to prove that the under-test product had achieved a level of stability conducive to the required parameters.

5. Conclusion

The integration of operational activities and flows at the level of an organization under a foreign information security policy is a must-have of every institution that wants digitalization. The system proposed and described in this paper can be customizable for each organization and allows individualization depending on future needs. Because it uses open-source platforms, as well as a cleverness of modularization in all its aspects, the system can be used by any organization and adapted to its requirements. For ease of use manuals for using modules, for system administration as well for its maintenance have been developed. They come to the aid of the beneficiary's administration and maintenance team, as well as of the system users. A future paper will present the elements of administration and maintenance of the system, as well as the possibilities for it to be accessed by the interested organizations.

REFERENCES

DOCKER (n.d.). Available at: <https://www.docker.com/>.

GeoServer (n.d.(a)). Available at: <http://geoserver.org/>.

GeoServer (n.d.(b)). *WMS - Web Map Service*. Available at: <http://docs.geoserver.org/stable/en/user/services/wms/index.html>.

GitLab (n.d.). Available at: <https://about.gitlab.com/>.

Han, Y. & Sun, R. (2016). Research on public management efficiency improvement method based on parallel database oriented optimization management information system, *RISTI Revista Iberica de Sistemas e Tecnologias de Informacao*, *E5*, 425-437.

HAProxy (n.d.). Available at: <http://www.haproxy.org/>.

InfoSecAddicts (n.d.). *Security Onion Components*. Available at: <https://infosecaddicts.com/security-onion-components/>.

Jones, S., Irani, Z., Sivarajah, U. & Love, E. P. (2017). Risks and rewards of cloud computing in the UK public sector: A reflection on three Organisational case studies, *Information Systems Frontiers*, *21*(2), 359-382.

JSON (n.d.). Available at: <https://www.json.org/json.html>.

Mohamed, N., Mahadi, B., Miskon, S., Haghshenas, H. & Adnan, H. M. (2013). Information System Integration: A Review of Literature and a Case Analysis. In Neck, R. (ed.), *Mathematics and Computers in Contemporary Science*, 68-77. World Scientific and Engineering Academy and Society (WSEAS).

NGINX (n.d.). Available at: <https://www.nginx.com>.

Postgres SGBD (n.d.). Available at: <https://www.postgresql.org>.

Praeg, C. P. & Spath, D. (ed.), (2010). Quality *Management for IT Services: Perspectives on Business and Process Performance*. IGI Global.

Report Server Community (n.d.). Available at: <https://reportserver.net/>.

Stephens, D. O. (2007). *Records Management: Making the Transition from Paper to Electronic*. Overland Park, KS: ARMA International.

Stock, G. N., Greis, N. P. & Kasarda, J. D. (1998). Logistics, strategy and structure: a conceptual framework, *International Journal of Operations & Production Management, 18*(1), 37-52.

WSO2 Application Server (n.d.). Available at: <https://wso2.com/products/application-server/>.

WSO2 Business Activity Monitor (n.d.). *WSO2 Business Activity Monitor Documentation*. Available at: <https://docs.wso2.com/display/BAM230/>.

WSO2 Business Process Server (n.d.). *WSO2 Business Process Server Documentation*. Available at: <https://docs.wso2.com/display/BPS360/>.

WSO2 Developer Studio (n.d.). Available at: <https://wso2.com/products/developer-studio/>.

WSO2 Enterprise Integrator (n.d.). *WSO2 Enterprise Integrator Analytics*. Available at: <https://docs.wso2.com/display/EI611/WSO2+Enterprise+Integrator+Analytics>.

WSO2 Enterprise Service Bus (n.d.). Available at: <https://wso2.com/products/enterprise-service-bus/>

WSO2 Governance Registry (n.d.). Available at: <https://wso2.com/products/governance-registry/>.

WSO2 Identity Server (n.d.). *Deliver Seamless Login Experiences with WSO2 Identity Server.* Available at: <https:// wso2.com/identity-and-access-management>.

WSO2 Message Broke (n.d.) *WSO2 Message Broker Documentation*. Available at: <https://docs.wso2.com/display/MB320/>.

ZABBIX (n.d.). Available at: <https://www.zabbix.com>.

Zhang, Y., Qiu, M., Tsai, C. W., Hassan, M. M. & Alamri, A. (2017). Health-CPS: Healthcare cyber-physical system assisted by cloud and big data, *IEEE Systems Journal*, *11*(1), 88-95.