ERP Tegia Components Update

Jeudi 04 Juin 2026

*Le monde est tel que nous le façonnons

The world is how we shape it*

sopra steria

Contexte Tegia

- Editeur du logiciel Tegia pour bailleurs sociaux (immobilier)
- 70 clients on premise
  - prod/test/projet/...
- Composants techniques complexes à mettre à jour (nécessite intervention humaine)
- Collision projets métiers à haute bande passante (facturation electronique)
- Besoin d'une solution "simple fiable robuste"
  - Impacts reglementaires (cyber resilience act)
- Tegia4 : 12.2 / tomcat9 / jdk11 / jre11
- Tegia5 : 12.8 / tomcat11 / jdk21(+17) / jre21

Sommaire

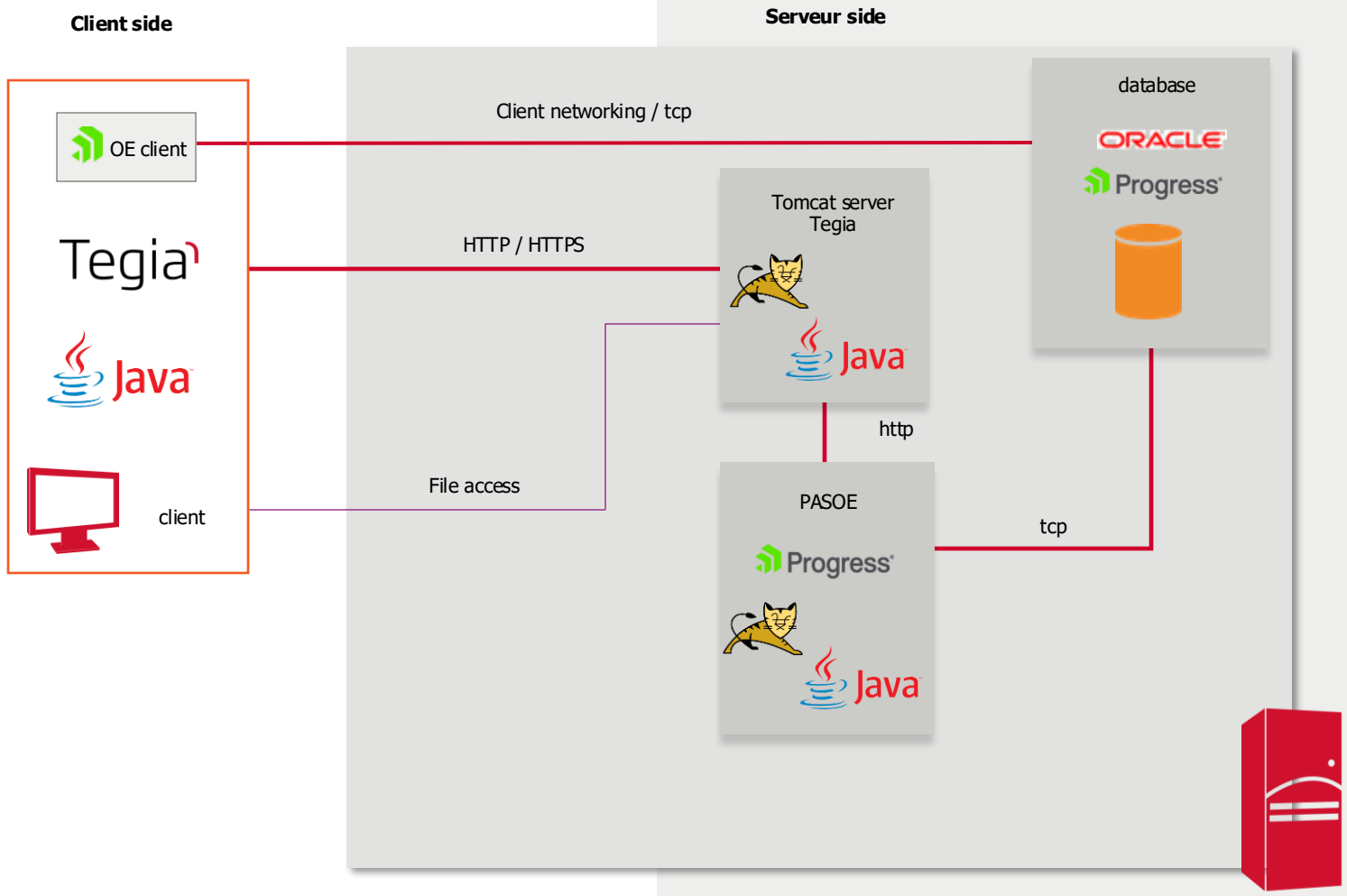
sopra steria

01

ERP Tegia Architecture

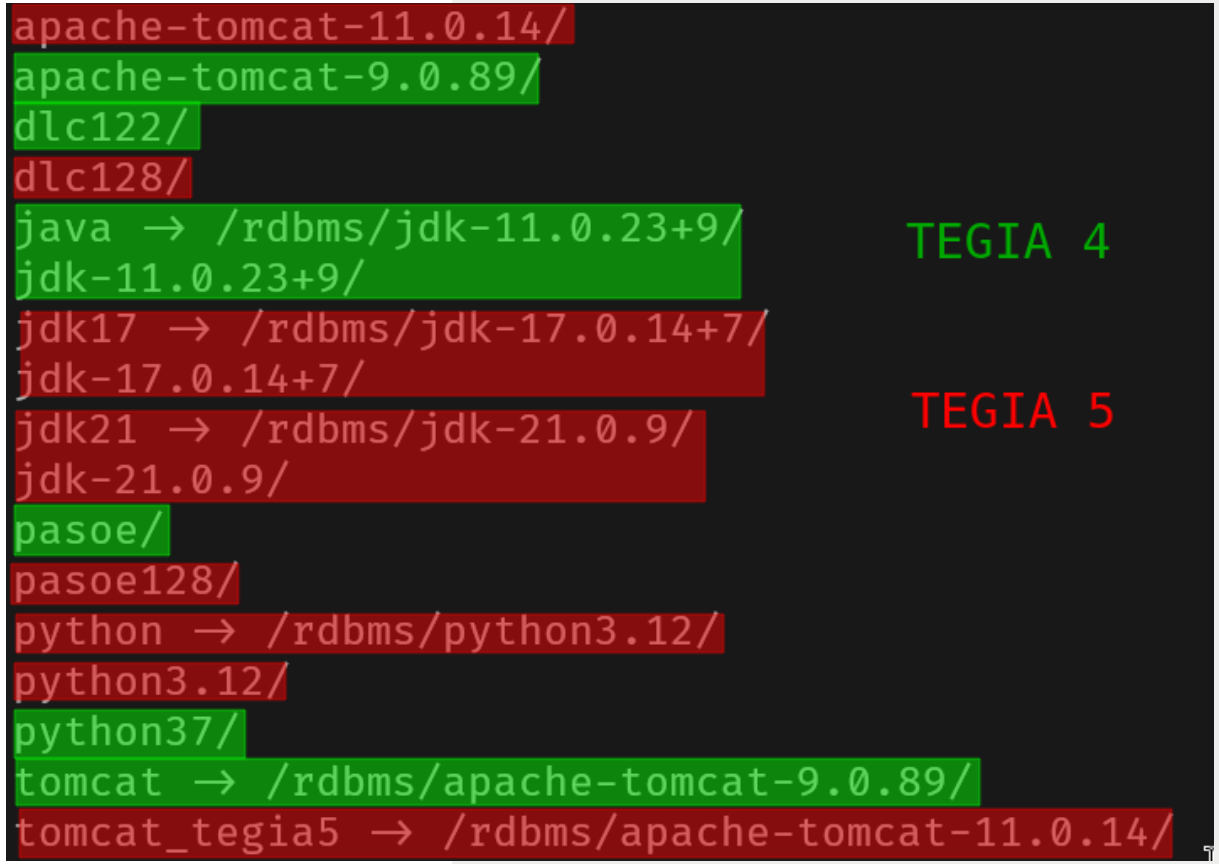
ERP Tegia Architecture

- Technical components
  - Server side
    - OpenEdge 12.8
    - tomcat 11
    - jdk 17 (for OE)
    - jdk 21 (for tomcat)
  - Client side
    - OpenEdge client (32bit runtime)
    - jre 21



Update Challenges

- Strong binding with OS
  - OpenEdge installation
    - Licences (products installation)
    - Adminservice
    - Properties configuration (conmgr/AdminServerPlugins/..)
    - PASOE deployment
- Same Host installation/deployment
  - Duplicate OE versions
    - Defaults adminservice configuration conflicts
  - Duplicate Tomcat/JDK/PASOE
    - Switch via environment variable
    - Per database settings
- Potential production environment running alongside !

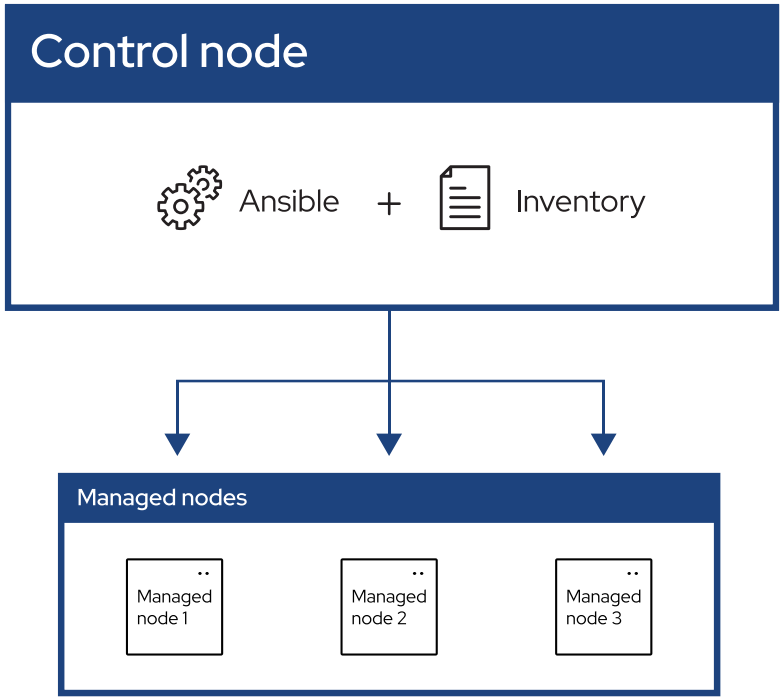


Tegia4/5 coexistence

Ansible Introduction

Ansible Overview

- Inventory management
  - Automate actions
  - Desired state
  - Idempotence / predictability
  - Declarative
  - IaC
  - Remote
  - Agentless ("nothing" required on the target system)
  - Single host vs fleet management
  - OSS and Redhat owned
  - Low controller requirements
    - Python + ssh access to target (winRM for windows)



Ansible Concepts

- Playbooks
  - Ansible code locations
  - Declaratives yaml files
- Tasks
  - Single instruction to be performed by the target system
- Modules
  - Code librairies
- Roles
  - Bundle of variables/tasks/modules/templates (independent or linked)
- Inventory
  - Node list
    - IP address/hostnames
    - Specific variables per host etc..

Ansible Concepts - playbooks

- Hosts
  - Explicitely defined in inventory
- Tasks
  - Name
  - Module
    - Parameters
- Desired state definition
  - Modules handle necessary steps
- Idempotence
  - Re-running the same tasks will yield similar outcome

```
- name: Update db servers
  hosts: databases
  remote_user: root

  tasks:
  - name: Ensure postgresql is at the latest version
    ansible.builtin.yum:
      name: postgresql
      state: latest

  - name: Ensure that postgresql is started
    ansible.builtin.service:
      name: postgresql
      state: started
```

Ansible Concepts - roles

- Grouping elements into roles
  - Tasks
  - Vars
  - Templates
  - Files
  - Handlers..

- Designed for portability and sharing

- Large roles collections by the community
  - ansible-galaxy

```
roles/
    common/               # this hierarchy represents a "role"
        tasks/            #
            main.yml      #  <-- tasks file
        handlers/         #
            main.yml      #  <-- handlers file
        templates/        #  <-- files for use with the template resource
            ntp.conf.j2   #  <------- templates end in .j2
        files/            #
            bar.txt       #  <-- files for use with the copy resource
            foo.sh        #  <-- script files for use with the script
resource
        vars/             #
            main.yml      #  <-- variables associated with this role
        meta/             #
```

Ansible Concepts - roles

```
# roles/example/tasks/main.yml
- name: Install the correct web server for RHEL
  ansible.builtin.import_tasks: redhat.yml
  when: ansible_facts['os_family']|lower == 'redhat'

- name: Install the correct web server for Debian
  ansible.builtin.import_tasks: debian.yml
  when: ansible_facts['os_family']|lower == 'debian'
```

```
roles/
    example/                # role example
        tasks/              #
            main.yml        #  <-- tasks file
            redhat.yml      # <-- tasks file
            debian.yml      # <-- tasks file
```

```
# roles/example/tasks/redhat.yml
- name: Install web server
  ansible.builtin.yum:
    name: "httpd"
    state: present
# roles/example/tasks/debian.yml
- name: Install web server
  ansible.builtin.apt:
    name: "apache2"
    state: present
```

Ansible Tegia Implementation

Tegia Updater

- Target functionality
  - Runned by a end user
  - Glorified script
  - Ansible idempotence
  - Benefit huge ansible collections/modules
  - Declarative
  - Bundled as a traditional tegia patch
- Package content
  - wrapper ansible script
  - User interactive script
  - Self contained python runtime
    - glibc requirement
    - All python required module dependencies included
  - Tegia updater playbooks

```
bootstrapper
├── ansible-wrapper.sh
├── res_tegia_updater/
├── tegia-updater.sh
└── test-winrm.sh

target
├── tegia-updater-packaged
│   ├── ansible_preliminary_setup.sh
│   ├── extrajob.automate
│   ├── python3.12.10-lnx64.tar.gz
│   ├── tegia-updater-bootstrapper.tar.gz
│   └── tegia-updater-playbook.tar.gz
└── tegia-updater-packaged.zip
```

Tegia Updater Playbooks

- Playbook layout
  - Split into individual subtasks
  - 3 entrypoints calling individual smaller playbooks
  - Custom filter plugin to handle both licenses files format
  - Templates
    - Silent install
    - PASOE openedge.properties

```
tegia-updater/
├── ansible.cfg
├── inventory/
│   └── hosts
├── playbooks/
│   └── tegia5/
│       │   └──{…}
│       └── vars_files/
│           ├── core_vars.yml
│           └── vars_file.yml
└── version_identifier
```

```
templates/
├── ch.properties.j2
├── ehcache.xml.j2
├── install_openedge_12.8.ini.j2
└── openedge.properties.j2
```

```
├── playbooks/
│   └── tegia5/
│       ├── common-def-win.yml
│       ├── common-def.yml
│       ├── filter_plugins/
│       │   └── parse_licenses.py
│       ├── init-system-deploy.yml
│       ├── migrate-env.yml
│       ├── update-system.yml
│       ├── resources/
│       ├── subtasks/
│       │   ├── check-prerequisites.yml
│       │   ├── deploy-pasoe-instance.yml
│       │   ├── install-oe-jdk.yml
│       │   ├── install-openedge-128.yml
│       │   ├── tailor-tomcat.yml
│       │   ├── update-cmd-scripts.yml
│       │   ├── update-dlc-win.yml
│       │   ├── update-jdk.yml
│       │   ├── update-jre-win.yml
│       │   ├── update-openedge-128.yml
│       │   ├── update-shell-scripts.yml
│       │   ├── update-tomcat.yml
│       │   └── validate-install.yml
│       └── templates/
```

Tegia Updater Templates

- Silent install ini template
  - Custom filter licenses plugin
    - Loop over the data stucture
  - Templated install paths
    - defaults value handling

```
[Configuration Count]
NumberofConfigurations={{ product_licenses|length }}



[Install Method]
ApplyAsServicePack=0

{% for license in product_licenses %}
[Product Configuration {{ loop.index }}]
name=PSC
serial={{ license.serial }}
version={{ license.version }}
control={{ license.control }}
prodname={{ license.prodname }}

{% endfor %}

[Java]
JavaHome={{ java_home_path|default('/rdbms/jdk17') }}

[Type and Destination]
type=COMPLETE
path={{ install_path|default('/rdbms/dlc128') }}
workpath={{ workpath|default('/rdbms/wrk128') }}
oem_path={{ oem_path|default('/rdbms/oemgmt128') }}
oem_workpath={{ oem_workpath|default('/rdbms/wrk_oemgmt128') }}
```

Tegia Updater Templates

- PASOE openedge.properties
  - Tailoring properties file for each target environment
    - Ports
    - PROPATH
    - PF location

```
[AppServer]
allowRuntimeUpdates=0
applications={{ target_environment }},{{ target_environment }}_apps

{…}

[AppServer.SessMgr.{{ target_environment }}]
agentLogFile=${catalina.base}/logs/{{ target_environment }}.agent.{yyyy-mm-dd}.log

[AppServer.Agent.{{ target_environment }}]
uuid=http://localhost:{{ http_port }}/{{ target_environment }}

[{{ target_environment }}_apps]
webApps={{ target_environment }}_app

[{{ target_environment }}_apps.{{ target_environment }}_apps.APSV]
adapterEnabled=1

{…}

PROPATH=${CATALINA_BASE}/webapps/{{ target_environment }}_apps/WEB-
INF/openedge,${CATALINA_BASE}/ablapps/{{ target_environment
}}_apps/openedge,${CATALINA_BASE}/openedge,${DLC}/tty,${DLC}/tty/netlib/OpenEdge.Net.apl,
${{"{"~target_environment~"PROPATH}"}}/pgjava,${{"{"~target_environment~"PROPATH}"}}/pgja
va/outils
```

Demo

Demo

https://asciinema.org/a/EQ1QYYOoD3uB3Nlt?speed=2&idleTimeLimit=2

https://asciinema.org/a/xEgeN7iwsrQCig4O?speed=2&idleTimeLimit=2

Conclusion

- 2 clients in production
  - Couple hours from start to production
  - Isolation between the two stacks
- Easier components update during tegia5 lifecycle
  - Only require user input to begin the update
  - Retrieve new components version and packages remotely
    - Simple static https webserver required
- Drawbacks
  - Around 30% full windows on premise installations

Q&A