



OCA Days

Liège 2025

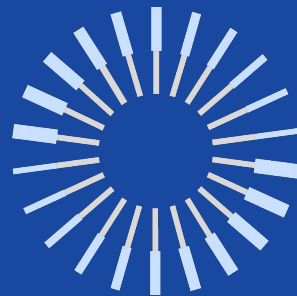
Making Odoo mightier, together

Multitenancy in Odoo: one instance to rule them all

Alexandre Nuttinck



DevOps Team Lead
at iMio



OCA Days
Liège 2025

Making Odoo mightier, together

Who am I?

- Alexandre Nuttinck
- DevOps Team Lead at iMio
- Based in Les Isnes, Wallonia, Belgium
- Discovered Odoo and OCA 2 years ago
- GitHub: github.com/alexnuttinck



iMio: Walloon intermunicipal association

- Provides IT services to ~400 local authorities
- Municipalities, CPAS, provinces, police zones, rescue zones
- 11 different applications → 1200+ instances
- Our mission: Mutualize IT solutions and support digitalization



Agenda

- The story of iA.Vision
- What is multitenancy?
- Why this presentation?
- How does it work in Odoo?
- Our architecture at iMio
- Benefits and drawbacks
- Q&A

The story of iA.Vision



Project Management Application for Public Administrations

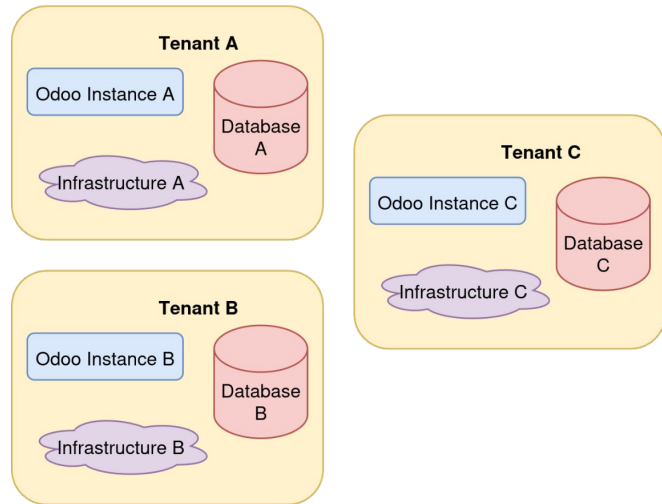
- Based on Odoo 16 (will be migrated to 18 next month)
- Operational management of projects and Cross-Strategic Plans
- Spoiler 🙌: **Currently serving 78 tenants** (local authorities)
- Open source: GPL3 License



OCA Days
Liège 2025
Making Odoo mightier, together

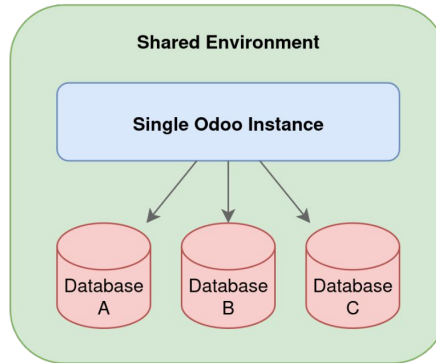
What is multitenancy?

Single Tenancy



VS

Multitenancy



One application instance serves multiple tenants

Shared infrastructure, codebase, and resources

Logical isolation between customers

Each tenant has independent data and configuration

Contrast: Single-tenant = separate instances per customer



The Multitenancy Decision

Key Questions We Faced

- Single instance vs. multiple instances?
- Many local authorities to serve
- Budget constraints of public sector, mutualization is in our DNA.
- Maintenance and update complexity

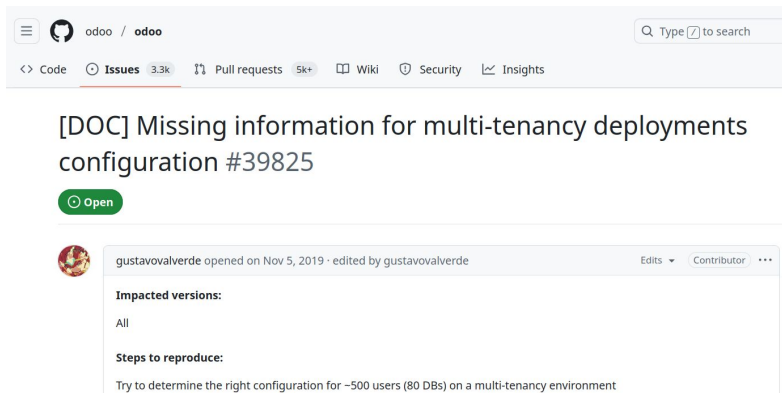
Decision: Multitenancy 



Why this presentation?

The Documentation Gap

- Limited Odoo multitenancy documentation
- Only `--db-filter` parameter documented
- Open GitHub issue since 2019: [#39825](#)
- Real-world examples are scarce
- Community needs practical guidance



OCA Days
Liège 2025
Making Odoo mightier, together

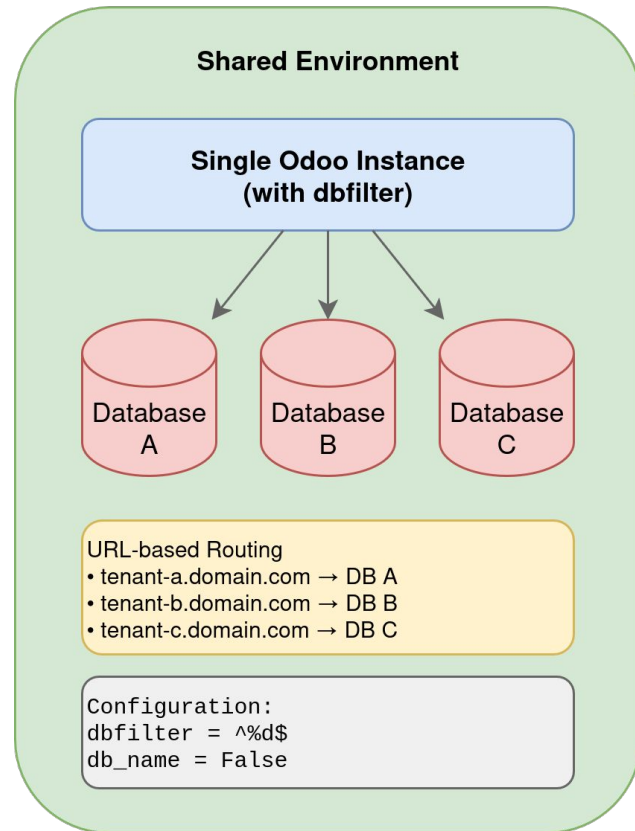
How does multitenancy work in Odoo?

Native Multitenancy Support

- Multiple databases on single Odoo instance
- Each database = one tenant
- Database filtering via `dbfilter` parameter

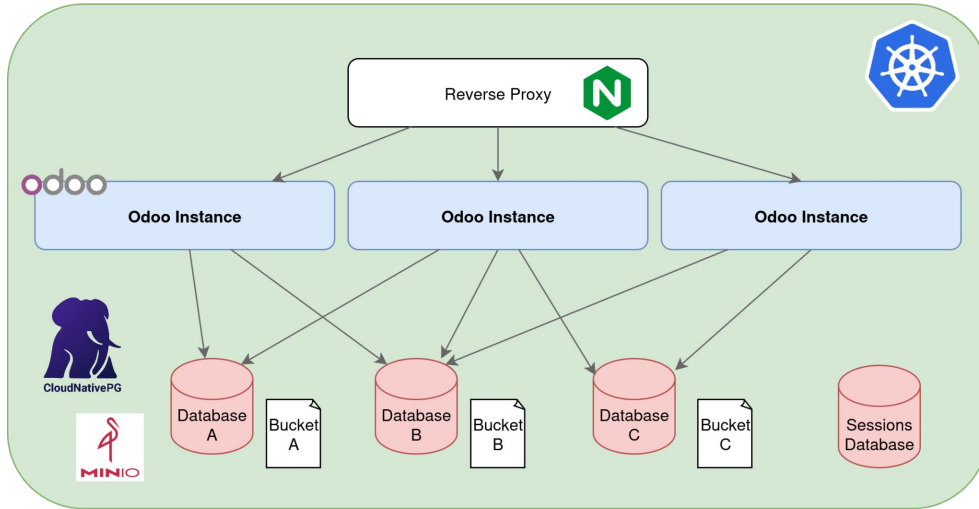
Key Configuration Elements

- Domain-based routing: `tenant-a.domain.be`
- OCA module that can help: [dbfilter_from_header](#)
- Memory management per worker
- Database connection limits
- Session handling across tenants



Our Architecture at iMio

iA.Vision architecture



Kubernetes: Container orchestration platform

Odoo: 3 replicas for high availability

NGINX: Reverse proxy, SSL termination, load balancing

PostgreSQL: CloudnativePG operator with PITR

S3 MinIO: Object storage for attachments (OCA [fs_storage](#) module)

Session storage: PostgreSQL-based sessions (OCA [sessions_db](#) module)

Current Odoo Configuration

```
replicas : 3
proxy_mode: True
limit_memory_soft: 2147483648
limit_memory_hard: 2684354560
limit_request: 65536
limit_time_cpu: 60
limit_time_real: 120
workers: 12
db_name: False
db_maxconn: 10
dbfilter: ^%d$
```

12 workers * 3 replicas

→36 simultaneous clicks

→216 connected users

We can change these parameters dynamically depending on the usage.

Source:

https://www.odoo.com/documentation/18.0/administration/on_premise/deploy.html#worker-number-calculation

Configuration : `dbfilter = ^%d\$`, `db\_name = False`

Memory Management : Set proper limits per worker

Database Connections : Monitor and limit
`db\_maxconn`

Session Storage : Use PostgreSQL, not filesystem

Monitoring : Prometheus + Grafana for observability



Benefits vs Challenges of Multitenancy

✓ Advantages We've Experienced

- **Economy of Scale:** Shared infrastructure reduces costs
- **Simplified Maintenance:** Single codebase to maintain
- **Easier Updates:** Deploy once, update all tenants
- **Resource Efficiency:** Better utilization of server resources
- **Faster Onboarding:** New tenants deploy quickly

⚠ Trade-offs We've Managed

- **Limited Customization:** Harder to meet specific tenant needs
- **Performance Risk:** Resource sharing can impact performance
- **Security Complexity:** Proper isolation is critical
- **Update Complexity:** Schema changes affect all databases
- **Troubleshooting:** Issues can be harder to isolate



Upgrade Strategy

Code-Only Updates (Zero Downtime)

- Rolling deployment with Kubernetes
- No database schema changes
- Instant rollback capability

Database Schema Updates

- Execution: Kubernetes Jobs with parallelism
- Safety: Maintenance mode
- Recovery: Point-in-Time Recovery for rollbacks

Our Numbers: ~15 minutes for 78 databases

```
apiVersion: batch/v1
kind: Job
metadata:
  name: "job-odoo-update"
  namespace: vision
spec:
  completions: 78
  parallelism: 10
  completionMode: Indexed
  ttlSecondsAfterFinished: 86400
  backoffLimitPerIndex: 0
  template:
    spec:
      initContainers: {...}
      containers:
        - name: update-db-odoo
          image:
            "harbor.imio.be/vision/vision:4.0.7"
          command:
            [
              "/bin/bash",
              "-c",
              'odoo -u all -d "$(cat
                /input/data.txt)"-vision
                --stop-after-init',
            ]
            {...}
          restartPolicy: Never
```



Should I use multitenancy for my use case? It depends 🤔

✓ Multitenancy is good for you if:

- Multiple similar customers/organizations
- Budget constraints require cost optimization
- Shared feature requirements across tenants
- Centralized maintenance is preferred
- Quick tenant onboarding is important

✗ Consider single-tenant if:

- Heavy customization per customer
- Strict isolation requirements (compliance, security)
- Unpredictable resource usage patterns
- Different Odoo versions per customer
- Complex integrations vary per tenant



Open Source Contributions

Giving Back to the Community

- Helm Chart: <https://github.com/IMIO/helm-odoo>
- iA.Vision: <https://gitlab.imio.be/odoo/ia.vision> (GPL3)
- Documentation: Real-world multitenancy examples
- Community Support: Sharing our experiences
- Sponsorship: Supporting OCA developments



What's Next for iA.Vision

Our Roadmap and Future Plans

- Odoo 18 Migration: Using OCA OpenUpgrade (planned next month)
- SSO Integration: Keycloak for centralized authentication
- Performance Optimization: Database read replicas, NGINX improvements
- Enhanced Monitoring: Direct metrics collection from Odoo instances
- Community Contributions: More open-source tooling and documentation



Thank you!

Questions & Discussion

✉ alexandre.nuttinck@imio.be

🐙 GitHub: [@alexnuttinck](https://github.com/alexnuttinck)

🌐 iMio: <https://www.imio.be>



Helm Chart: <https://github.com/IMIO/helm-odoo>



iA.Vision: <https://gitlab.imio.be/odoo/ia.vision>



OCA Days
Liège 2025
Making Odoo mightier, together

Thank you to our Event Sponsors



OCA Days
Liège 2025
Making Odoo mightier, together