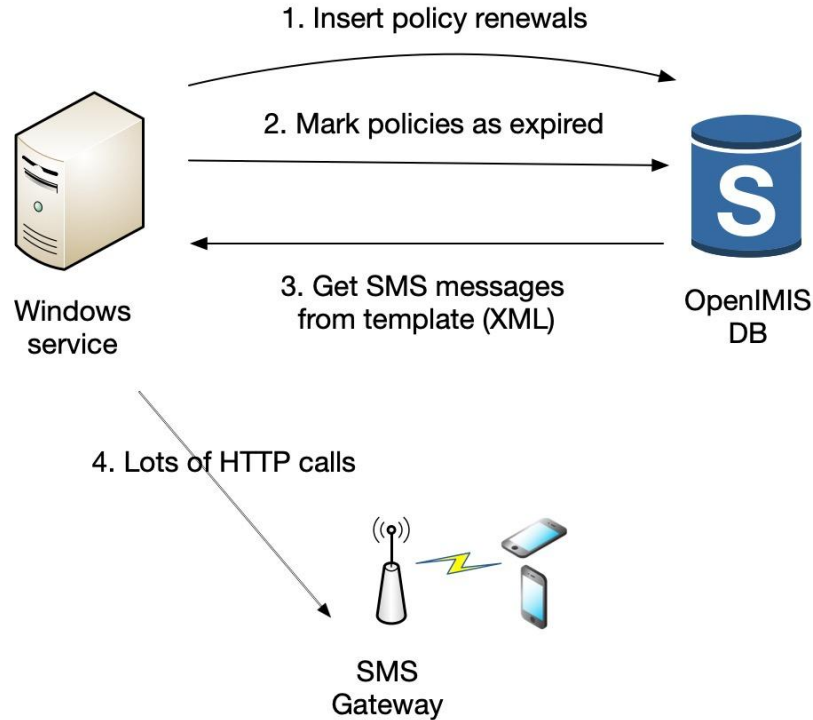


openIMIS

Scheduled Processing



openMIS - current scheduling



openMIS - scheduled processing

- Scheduled (or 'batch') processing in IT systems

You **always** want to keep most of your resources for your “**interactive users**”

... and thus keeping background stuff in the background (constraining resource consumption)

Example:

DON'T import 5M claims via the **current** openMIS FHIR API (we should be looking at <http://bit.ly/fhir-bulk-api> ;-))

- In the [initial roadmap](#), we proposed to use [Apache AirFlow](#)

- Open Source, very complete, 'python friendly',...

- **BUT:**

- it requires a separate docker - *that should be also the case for any solution in any prod env. (batch on their own resources)*
- it is 'yet another technology' (with dedicated vocabulary, 'UI style',...)

- In line with to our (togaf-based) methodology, we re-assess the use of AirFlow

- because it was not used (and thus setup) yet

- ... and there will (*should be*) other opportunities to change/improve/...



openIMIS - scheduled processing - [OMT-215](#)

Assessed solutions:

- OpenHIM (and OpenHIE ecosystem at large)
- OpenMRS (what did they do?)
- django existing modules for “scheduling/background tasks/...”



OpenHIM (thanks Daniel)

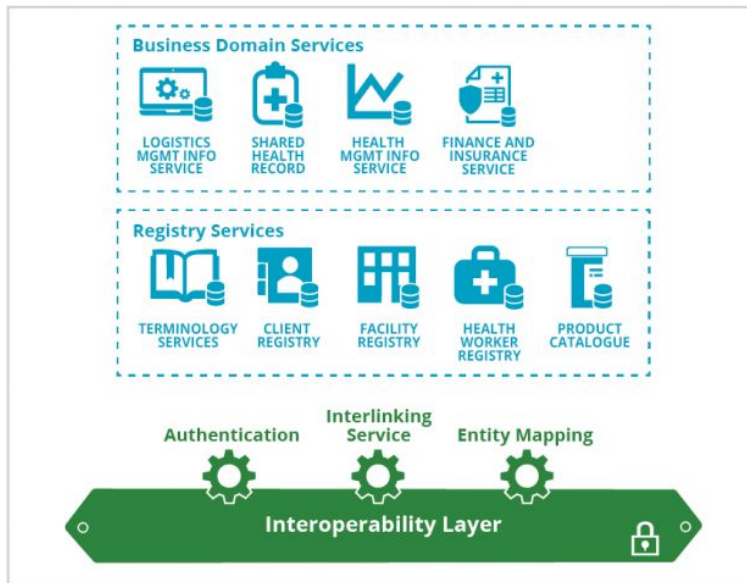
- Has some features ‘in scope’
 - Bulk data (‘large payload’ messages) processing
 - Polling channel
 - Orchestration capabilities
 - Monitoring (error replay,...)
 - ... could be customized via Mediator to encompass our needs
- **BUT**
 - OpenHIM as a rather distinct ‘angle’ (usage of these features): it is focus on “data exchange”
 - Is better positioned ‘down’ or ‘up’ stream of our scheduled processes
(example: sending SMS based on our prepared communication in our policy renewal scheduled process)
 - ... but not so well for openIMIS ‘internal’ processing
(trigger and follow-up of our policy renewal batch)
- **THUS**

Although integration with OpenHIM is relevant in our scheduled processes scope (as it is in all our data exchanges with other systems), OpenHIM is not well suited to be our “foundation” for scheduled processing in openIMIS



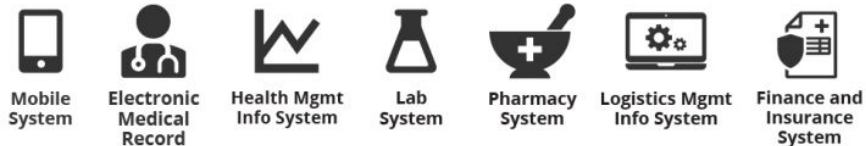
OpenHIE

OpenHIE
Component Layer



Interoperability
Services Layer

Point of Service



OpenHIE 2020-05-28; CC BY 4.0

Question to OpenHIE community:

Would an 'central scheduler' fit in the Interoperability Services Layer?



OpenMRS

OpenMRS has integrated scheduled task management

<https://wiki.openmrs.org/display/docs/Administering+Scheduled+Tasks>

The screenshot shows a web browser displaying the OpenMRS Wiki page for 'Administering Scheduled Tasks'. The page content includes:

- Administrator Guide**
- Administering Scheduled Tasks**
- Scheduled Tasks are regularly timed tasks that can run every few seconds, every day, every week, etc. See Admin-Manager Scheduled Tasks for the administration of them.
- A task is just a Java class that is registered in OpenMRS by being in the `schedulerTask_config` table. Typically Modules provide these tasks.
- You can edit the properties of a task by clicking through to it.
- Prior to v1.7 of OpenMRS you had to set a `username/password` used to run tasks in your `openmrs` runtime properties file. This lets the tasks authenticate to openmrs to do their things. (After 1.7 you can delete this entry because tasks run as the 'root' user by default).
- There are a number of default tasks provided:
 - Process Form Entry Queue - used by the formentry to process xml into hl7
 - Process HL7 Task - reads the HL7 in Queue table and turns them into rows in encounter/obs/patient/etc
 - Alert FormEntry Task
 - Send Email Task
 - Health World Task
 - Check Internet Connectivity Task - tries to reach google.com and adds an `isInt`
 - Update Concept Words - refreshes all concept names (see Admin → Update Concept Words page)
 - Generate Data Export - takes in a comma separated 'start&enddate' argument which specifies which Data Exports (reporting/compatibility module) to generate
 - org.openmrs.scheduler.tasks.GenerateDataExportTask

... developed 'custom' inside OpenMRS code

The screenshot shows the GitHub repository for `openmrs-core`. The repository structure is as follows:

- Branch: `master`
- Path: `openmrs-core / api / src / main / java / org / openmrs / scheduler /`
- Files and Commits:
 - `db`: Trunk-5710: Implement the getByUniqueId(String uniqueid) in SchedulerS... (3 months ago)
 - `tasks`: GCI-142 Lambdas and Method references instead of anonymous classes (#... (3 years ago)
 - `timer`: TRUNK-5706: Replace all instances of '@should' (3 months ago)
 - `Schedule.java`: TRUNK-4917 - Updated domain objects to extend the new base classes fo... (3 years ago)
 - `SchedulerConstants.java`: TRUNK-5033 Public static fields should be final, made the appropriate... (3 years ago)
 - `SchedulerException.java`: Converted to MPL 2.0 HD and standard line endings (5 years ago)
 - `SchedulerService.java`: TRUNK-5706: Replace all instances of '@should' (3 months ago)
 - `SchedulerUtil.java`: TRUNK-5706: Replace all instances of '@should' (3 months ago)
 - `StatefulTask.java`: Converted to MPL 2.0 HD and standard line endings (5 years ago)
 - `Task.java`: Converted to MPL 2.0 HD and standard line endings (5 years ago)
 - `TaskDefinition.java`: GCI-142 Changed all Loggers to private static final & remove unuseds ... (3 years ago)
 - `TaskFactory.java`: TRUNK-5437 Use parametrized messages while logging. (#2722) (2 years ago)

... we are not going to deploy OpenMRS along openIMIS (at least not 'just' for scheduling)



Django existing modules: Celery

- ✓ Already part of the platform
- ✓ Easy to setup and configure

```
CELERYBEAT_SCHEDULE = {  
    'policy_renewals_mon': {  
        'task': 'policy_renewals',  
        'schedule': crontab(hour=6, minute=0, day_of_week='monday'),
```

- ✗ No concurrency management
- ✗ No pause/resume, check of completion
- ✗ No administration interface



Django existing modules: django-cron

- Only handles the concurrency of jobs and similar issues
- Relies on a Django command started from crontab
- Designed as stand-alone



Django existing modules: django-background-tasks

- ✗ More of a “delayed task” than a real batch platform
- ✗ No update in over 6 months, compatible with Django 2.2. => 3?
- ✗ Execution of the tasks relies on running a manage.py command



Django existing modules: django-scheduler

✓ Great calendar UI

- ✗ Tricky integration into current UI build
- ✗ Calendar UI is cool not so useful
- ✗ No concurrency management

Example Calendar
September 2015
This month

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
		1 18:36 Hola	2 (All day) Hola	3 (All day) Hola	4 (All day) Hola	5 (All day) Hola
6 (All day) Hola	7 (All day) Hola	8 (All day) Hola	9 (All day) Hola	10 (All day) Hola	11 (All day) Hola	12 (All day) Hola
13 (All day) Hola	14 (All day) Hola	15 (All day) Hola	16 (All day) Hola	17 (All day) Hola	18 (All day) Hola	19 (All day) Hola
20 (All day) Hola	21 (All day) Hola	22 (All day) Hola	23 (All day) Hola	24 (All day) Hola	25 (All day) Hola	26 (All day) Hola
27 (All day) Hola	28 (All day) Hola	29 (All day) Hola	30 (All day) Hola			

Three Month Calendar Full Year Calendar



Django existing modules: APScheduler

Advanced Python Scheduler

- ✓ Can run with its own process or be bundled into the Django instance (easy)
- ✓ Manages scheduling and concurrency
- ✓ Integrated into the Django Admin console
- ✓ Jobs can be started/paused/cancelled from the console
- ✓ Job completion or failure is recorded
- ✓ Uses the database rather than RabbitMQ or Redis
- ⚠ Works with Django (module available) but with some caveats



Django existing modules: APScheduler

Select django job execution to change

Action: 0 of 100 selected

<input type="checkbox"/>	ID	Job	Html status	Run time	Duration
<input type="checkbox"/>	1810	app.management.commands.run_scheduler.test (next run at: 2017-05-19 18:27:41.138072)	Executed	May 19, 2017, 6:27 p.m.	1.15
<input type="checkbox"/>	1809	app.management.commands.run_scheduler.test (next run at: 2017-05-19 18:27:41.138072)	Error!	May 19, 2017, 6:27 p.m.	1.15
<input type="checkbox"/>	1808	app.management.commands.run_scheduler.test (next run at: 2017-05-19 18:27:41.138072)	Executed	May 19, 2017, 6:27 p.m.	1.14
<input type="checkbox"/>	1807	app.management.commands.run_scheduler.test (next run at: 2017-05-19 18:27:41.138072)	Error!	May 19, 2017, 6:27 p.m.	1.14
<input type="checkbox"/>	1806	app.management.commands.run_scheduler.test (next run at: 2017-05-19 18:27:41.138072)	Executed	May 19, 2017, 6:27 p.m.	1.14
<input type="checkbox"/>	1805	app.management.commands.run_scheduler.test (next run at: 2017-05-19 18:27:41.138072)	Error!	May 19, 2017, 6:27 p.m.	10.46
<input type="checkbox"/>	1804	app.management.commands.run_scheduler.monitor (next run at: 2017-05-19 18:32:01.062115)	Executed	May 19, 2017, 6:27 p.m.	10.11
<input type="checkbox"/>	1803	app.management.commands.run_scheduler.file_stats (next run at: 2017-05-19 18:32:00.986196)	Executed	May 19, 2017, 6:27 p.m.	9.51
<input type="checkbox"/>	1802	app.management.commands.run_scheduler.test (next run at: 2017-05-19 18:27:41.138072)	Error!	May 19, 2017, 6:26 p.m.	1.14
<input type="checkbox"/>	1801	app.management.commands.run_scheduler.test (next run at: 2017-05-19 18:27:41.138072)	Error!	May 19, 2017, 6:26 p.m.	1.14
<input type="checkbox"/>	1800	app.management.commands.run_scheduler.test (next run at: 2017-05-19 18:27:41.138072)	Error!	May 19, 2017, 6:26 p.m.	1.14
<input type="checkbox"/>	1799	app.management.commands.run_scheduler.test (next run at: 2017-05-19 18:27:41.138072)	Executed	May 19, 2017, 6:26 p.m.	1.14
<input type="checkbox"/>	1798	app.management.commands.run_scheduler.test (next run at: 2017-05-19 18:27:41.138072)	Executed	May 19, 2017, 6:26 p.m.	1.14
<input type="checkbox"/>	1797	app.management.commands.run_scheduler.test (next run at: 2017-05-19 18:27:41.138072)	Error!	May 19, 2017, 6:26 p.m.	1.15
<input type="checkbox"/>	1796	app.management.commands.run_scheduler.test (next run at: 2017-05-19 18:27:41.138072)	Error!	May 19, 2017, 6:26 p.m.	1.14
<input type="checkbox"/>	1795	app.management.commands.run_scheduler.test (next run at: 2017-05-19 18:27:41.138072)	Error!	May 19, 2017, 6:26 p.m.	1.14
<input type="checkbox"/>	1794	app.management.commands.run_scheduler.test (next run at: 2017-05-19 18:27:41.138072)	Executed	May 19, 2017, 6:26 p.m.	1.15



Conclusion and proposal

- For our current scope (Policy & insurree pictures renewal), AirFlow is by far overkill

(and would add a complexity to deployment stack at the wrong moment: legacy/new stack cohabitation makes it already very complex)
- OpenHIM (which could 'justify' additional complexity) is not well suited on several scheduled processes aspects
- ... best option is to use *APScheduler* (based on our needs for policy renewals)
 - trying to 'remain open' (encapsulate in REST API,...) to external scheduling, or bulk data exchanges
 - ... and re-assess the day we would need more