# A Modular Workflow Management Framework

João Rafael Almeida, Ricardo Ribeiro and José Luís Oliveira
*University of Aveiro, DETI/IEETA, Portugal*

Keywords: Task Management, Workflow Management, Clinical Studies.

Abstract: Task management systems are crucial tools in modern organizations, by simplifying the coordination of teams and their work. Those tools were developed mainly for task scheduling, assignment, follow-up and accountability. Then again, scientific workflow systems also appeared to help putting together a set of computational processes through the pipeline of inputs and outputs from each, creating in the end a more complex processing workflow. However, there is sometimes a lack of solutions that combine both manually operated tasks with automatic processes, in the same workflow system. In this paper, we present a web-based platform that incorporates some of the best functionalities of both systems, addressing the collaborative needs of a task manager with well-structured computational pipelines. The system is currently being used by a European consortium for the coordination of clinical studies.

## 1 INTRODUCTION

Time and resource management is one of the most important issues in organizations, groups and even at the individual level. It is especially relevant in the corporate environment, where tasks' duration may have a direct impact on a company's performance and its results. As such, task optimization has always been a key aspect for management teams. This process has been facilitated by the adoption of dedicated computer programs that helped simplify team and task management, by decomposing projects into tasks, which can be assigned, analysed, performed and refined, over time.

The evolution of software engineering solutions also opened the path for the appearance of microservices distributed architectures, e.g., SOA and Web services (Papazoglou, 2003; Sheng et al., 2014), which allow the construction of new applications and processing pipelines based on the reuse of existing services. This can be implemented through workflow management systems (Liu et al., 2015).

While workflow systems tend to focus on the relations between services and the execution pipeline, dismissing users and manually operated tasks, task management systems focus almost exclusively on the tasks and their assignees. Thus, the relation between tasks, their inputs/outputs, and how they relate to each other are disregarded. Besides, most existing software solutions tend to be problem-specific, being focused on a particular speciality. Thus, these solutions are usually very context-specific and unable to cope with a more generalised environment.

On the other hand, solutions such as business-process managers end up not being usable for several use cases, since they are too generic (vom Brocke et al., 2016), creating a layer of complexity that makes it difficult for the average user to utilize and comprehend. This shortcoming in systems favours the development of a solution that brings together the best of both task managers and workflow systems.

Despite the previous discussion of the broader scenario, the original motivation for this work was to contribute to simplifying the management of clinical research studies, based on multiple and heterogeneous Electronic Health records (EHR) (Oliveira et al., 2013; Gini et al., 2016). The quantity of clinical information and disease-specific data has been continuously increasing in the past years. This information is fragmented over dispersed databases in different clinical silos around the world. However, as awareness about the potential of these data for clinical research increases, there is a growing need for solutions for secure exploration of these data across different centres (Bastião et al., 2015). This re-use of data may lead to many health benefits, mainly for clinical and pharmacological researchers (Burgun et al., 2017). However, due mostly to ethical and legal issues, it is still very difficult to integrate these data into a single repository, or even to obtain access to



Almeida, J., Ribeiro, R. and Oliveira, J.
A Modular Workflow Management Framework.
DOI: 10.5220/0006583104140421
In *Proceedings of the 11th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2018) - Volume 5: HEALTHINF*, pages 414-421
ISBN: 978-989-758-281-3

them (Lopes et al., 2015). To overcome these challenges, researchers have to deal with complex processes that include study submission, governance approval, data harmonization, data extraction, statistical analysis and many other tasks. From this scenario, the need emerged for a task/workflow management system that simplifies execution of these processes, among many centres and users.

In this paper we describe a system that addresses the above-mentioned issues. It consists of a modular platform that allows collaboration between different users through a user-friendly web-based interface, while keeping a strong focus on the relation between the tasks that users perform. This system was developed in the context of EMIF (http://www.emif.eu), a European project that aims to create a common technical and governance framework to facilitate the reuse of health data.

## 2 BACKGROUND AND RELATED WORK

The use of a software application to manage projects, teams and tasks is not new. Indeed, this idea has been explored and has been progressively growing in several areas. For instance, managing business processes is crucial in any efficient organization, and as such, they naturally adopt or develop systematised solutions to manage those processes. On the other hand, a workflow is normally perceived as the orchestration of repeatable business processes that process information in a systematic fashion. Workflow management platforms allow us to re-engineer business and information processes, facilitating the flow of information, and notifying actors whenever their input is needed (Georgakopoulos et al., 1995).

In this section, we will analyse some of the task and workflow management solutions that are currently available, following two perspectives: 1) end-user applications, which are ready to use; and 2) software engines, that can be used to integrate in more complex solutions.

### 2.1 Fully-fledged Solutions

There are a large number of web and cloud-based ready-to-use solutions that fulfil part of our requirements. However, most of them are commercial. Moreover, they do not allow integration with other systems, so it is not possible to extend their interface to integrate with external applications.

Wrike[1], for instance, is a cloud-based collaborative platform, where users can assign tasks, and track deadlines and schedules. It follows the workflow concept and it allows integration with document management solutions, allowing use in project management and social cooperation.

Asana[2] is another cloud-based solution, targeted at project and task management, which can be helpful for teams that handle multiple projects at the same time, and it can serve teams of any size.

When seeking a scientific workflow management system, Taverna[3] takes the lead, among many others. This system is available as a suite of open-source tools to facilitate computer simulation of repeatable scientific experiments (Wolstencroft et al., 2013). It can be executed in a self-hosted server or as a desktop client, after proper installation. The system follows an SOA approach, which makes the various web interfaces available for external software integration. The learning curve seems steeper for new users than in other platforms, but it is widely adopted in scientific studies, namely in the bioinformatics area.

Another relevant tool in this domain is Galaxy (Afgan et al., 2016), a python-based platform aimed to facilitate computational biomedical research over big datasets. One of the main goals of this platform is to be easily used by those without technical knowledge. It allows the repetition and sharing of studies, the interface is concise with a good visual editor, but the tasks defined in the workflow belong typically to a restricted domain.

### 2.2 Workflow Engines

Conceptually, workflow engines only manage the automated aspects of a workflow process for each item, determining which activities are executed, and next, when pre-requisites are achieved, attributing these tasks. The idea of these tool-kits is that they can be customised, integrated and extended in larger software projects. A workflow management engine does not offer a ready-to-use solution, but only the base blocks to build a new system. Although this brings the obvious disadvantage of having to develop the end-user system, it also brings several advantages, mainly due to the flexibility to integrate other software modules.

Activity[4] is a lightweight workflow management platform focused on the needs of business professionals, developers and system administrators. This plat-

[1]https://www.wrike.com
[2]https://www.asana.com
[3]https://taverna.incubator.apache.org
[4]https://www.activiti.org/

form allows complex repeatable workflows with different kinds of tasks, but with only one assignee at a time, even though it allows reassignments in the middle of a process.

FireWorks[5] is another open-source project, which is focused on the management and execution of scientific workflows. Most of the system focuses on parallel work execution and on job scripting and processing, even having integration with popular task queuing platforms.

GoFlow[6] is a python-based workflow engine that is provided as a module component for the Django framework. The engine is activity-based, allowing the specification of a flow between activities, distributed to different users. Although this fulfils the basic needs of our project, and seems easy to integrate in a more complex solution, it has the main problem of not allowing background tasks by default.

jBPM[7] is an open-source business process management suite, embedded in the KIE group, which executes repeatable workflows. This solution is Java EE based and it runs as a Java EE application. The system supports multi-user collaboration, using groups of users, but its configuration is rather complex for users without technical skills.

Current fully-fledged web platforms lack essential features such as allowing asynchronous tasks and easy integration with external tools. Furthermore, existing workflow engines do not support multi-user features such as easy collaboration over the same workflows, discussion of collected results, and workflow sharing between different users, greatly impairing collaboration efforts.

Therefore, we decided to build a new web platform that allows easy collaboration between partners, with multi-user interactions and features such as result discussion and workflow sharing.

## 3 SYSTEM REQUIREMENTS

From the previous section, there is seen to be a wide range of solutions in this field. However, as previously discussed, it is hard to find a solution that combines the potential of a task and workflow management system, i.e. allows defining workflows that mix computational processes with human-oriented tasks.

To build this specific solution, we needed a carefully planned set of requirements that allow fulfilment of users' needs. A core idea guiding this development was that any user should be able to work easily with the system, without needing to be an expert in task or workflow management systems. Moreover, we also envisaged adopting the best and most updated software engineering practices, to ensure the system's modularity and extensibility.

Building on our previous experience in observational data projects, and in close collaboration with EMIF partners and potential users, we devised a set of functional requirements that guided the system development from the beginning, although some others were incorporated along the way, following an agile methodology. Here, we describe some key ones:

- The user interaction must be performed entirely through an HTML5 web interface;
- Allow management of users, activities and roles, using RBAC policies (Ferraiolo et al., 1995);
- Private workspace for each user to manage their assigned tasks. Here, it should also be possible to create tasks and workflows;
- Different types of tasks, namely *manual*, *questionnaire*, and *service*;
- The workflow may be private (only the owner can edit and execute), or public (can be executed, or copied for further refinements);
- The user should be able to duplicate a workflow and edit the copy as they wish without affecting the original workflow;
- Creation of workflows combining any sequence of tasks, pipelining the previous outputs to the following tasks' inputs;
- Workflows can be started on any given date, and repeated, with different deadlines;
- Workflow tasks may be assigned to different actors and with different deadlines and requirements;
- All users involved in the study should also be able to give feedback about their own tasks;
- The manager should be able to ask for refinements and to reassign tasks to other users;
- The system needs to notify users about task deadlines and progress;
- The workflow manager (the one who started the workflow) must be able to follow and share the pipeline execution;
- The system must follow a service-oriented architecture, based on REST web services, so that it can be easily reused in multiple platforms, as a server engine. This implies that, besides the web

[5] https://github.com/materialsproject/fireworks
[6] https://goflow.me/
[7] http://www.jbpm.org/

interface for end-users, it must be able to be executed entirely through REST services.

Several non-functional requirements were also defined, namely cross-platform deployment, modularity and portability.

# 4 SYSTEM ARCHITECTURE

To address the initial requirements and to obtain high maintainability and scalability, we developed our task/workflow management system in a two-layered architecture. The decision for a default REST engine aimed for the possibility of partial or full integration in other clients in the future. We followed the micro-kernel pattern (Richards, 2015) to allow easy incorporation of new types of tasks.

In the following sub-sections we will detail the technological approach. Firstly, from the backend engine perspective (*Backend Core*) which ensures the application's business logic and works independently from the others. Secondly, from the frontend client perspective (*Web Client Core*), built upon HTML5 frameworks and relying on the backend web services. And finally, from a deployment perspective, the attempts made to ensure effortless installation of new instances in all systems.

Figure 1 presents a generalised view of the overarching architecture components for the system environment.

## 4.1 Backend Engine

This sub-section describes the technologies and architecture of the server, including the services provided through the web services API.

### 4.1.1 Technologies

The server was developed in python language using the Django framework[8]. For the web services, we also used the Django REST API, a powerful and flexible library built over Django.

To support asynchronous jobs, namely modules which have long execution times, we used Celery[9] and RabbitMQ[10] . Celery supports direct integration with Django, which made it the perfect fit for handling background processes in our system, such as scheduled actions and long-running events like sending notifications through emails.

The representation of data in Excel spreadsheets simplifies its analysis, because it has more advanced features than text and CSV files. The change of task results to this format is handled through Openpyxl[11].

For data persistence we used PostgreSQL[12], together with Django object-relational mapping (ORM). Finally, for error tracking we rely on Sentry[13], since it provides good mechanisms for logging and analysis.

### 4.1.2 Architecture

One of the technical goals was to keep the application modular and extensible, reducing the core to the minimum possible. As such, the server follows the layered architecture pattern, mixed with the micro-kernel pattern for including new types of tasks, results and resources.

The final server organization is depicted in Figure 2 where we can see how components relate with each other, namely the modules that provide services and the modules accepting core extensions through new applications, without having to interact with the rest of the system. In this diagram, *History* works as a transversal component that serves all the other modules, being responsible for recording all the actions and events that occur in the backend. The system resources are managed through the *Material* component, which also moderates the allocation of resources required by the other modules. The *Result* is responsible for handling the different types of results that can be generated by each task. The *Process* and *Workflow* components are inter-related and responsible for the workflow instances management (i.e. study templates and running instances).

As already mentioned, the system is able to work only using REST web services. All of them are JSON based, except file uploads which are handled as binary data.

## 4.2 Frontend Client

This sub-section will contain a detailed explanation of all technologies used for building the frontend default client which consume the backend web services and display the information in a pleasant, easy-to-use interface, which makes workflow management available by visual interaction.

[8]https://www.djangoproject.com
[9]http://www.celeryproject.org
[10]https://www.rabbitmq.com
[11]https://openpyxl.readthedocs.io/
[12]https://www.postgresql.org
[13]https://sentry.io/

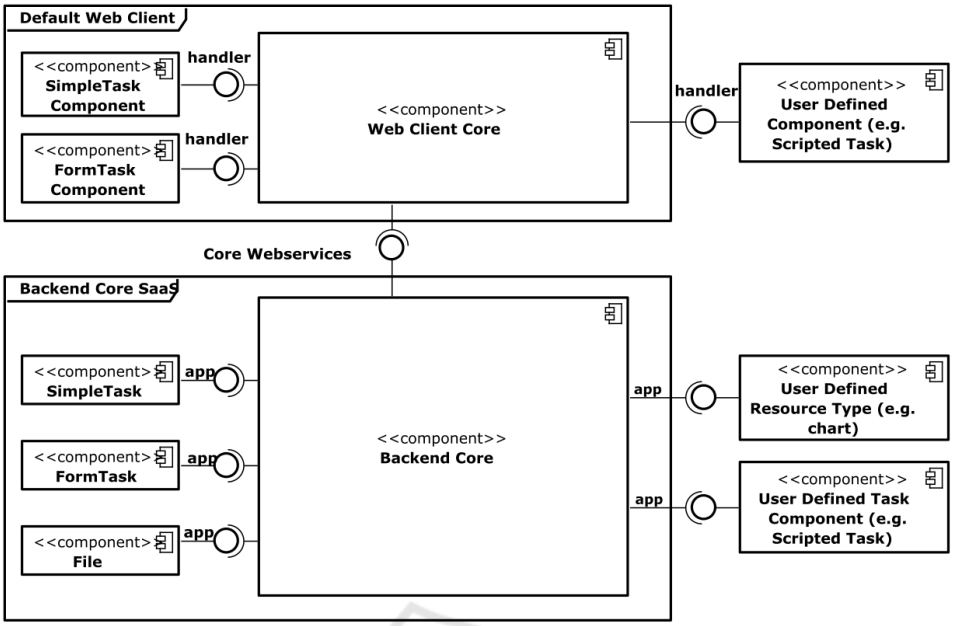


Figure 1: General architecture.

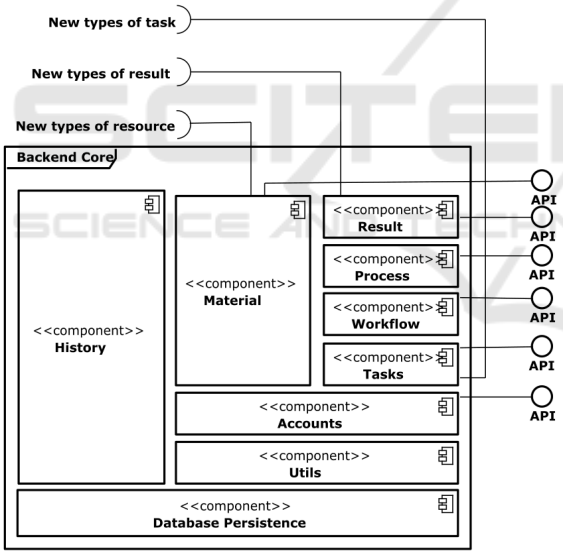


Figure 2: Backend Architecture.

#### 4.2.1 Technologies

To develop the front-end of the application, we use ReactJS[14] as the basis of our solution. Some aspects influenced this choice, namely the very active community and successive improvements over time, and also allowing faster development cycles.

As the solution was based on consuming backend web services, it was necessary to combine ReactJS with another technology. For this, we used Reflux[15], a uni-directional dataflow application toolkit.

For the frontend, a critical aspect is the layout structure and web appearance. We decide to rely on Bootstrap[16] a popular open-source solution framework for layout development. Still, some components had to be developed since they were not included in the default Bootstrap package.

Finally, we developed our own workflow edition schema, a key piece of the final solution, since we could not find a good enough open-source solution to address the requirements. This workflow editing scheme supports the creation and editing of study templates through a visually attractive and intuitive interface.

In Figure 3, it is possible to visualize the edition of a study already constituted with several tasks. It is possible to see a panel on the left side, which contains all the information related to the selected task. On the right, the workflow is represented with its tasks and the relationships between them.

#### 4.2.2 Architecture

The diagram presented in Figure 4 is a resumed version of the frontend architecture, its components and the way they communicate with each other.

Similarly to the backend core, the client side was designed as a modular and extensible solution, where

[14]https://github.com/reactjs

[15]https://github.com/reflux/refluxjs

[16]http://getbootstrap.com

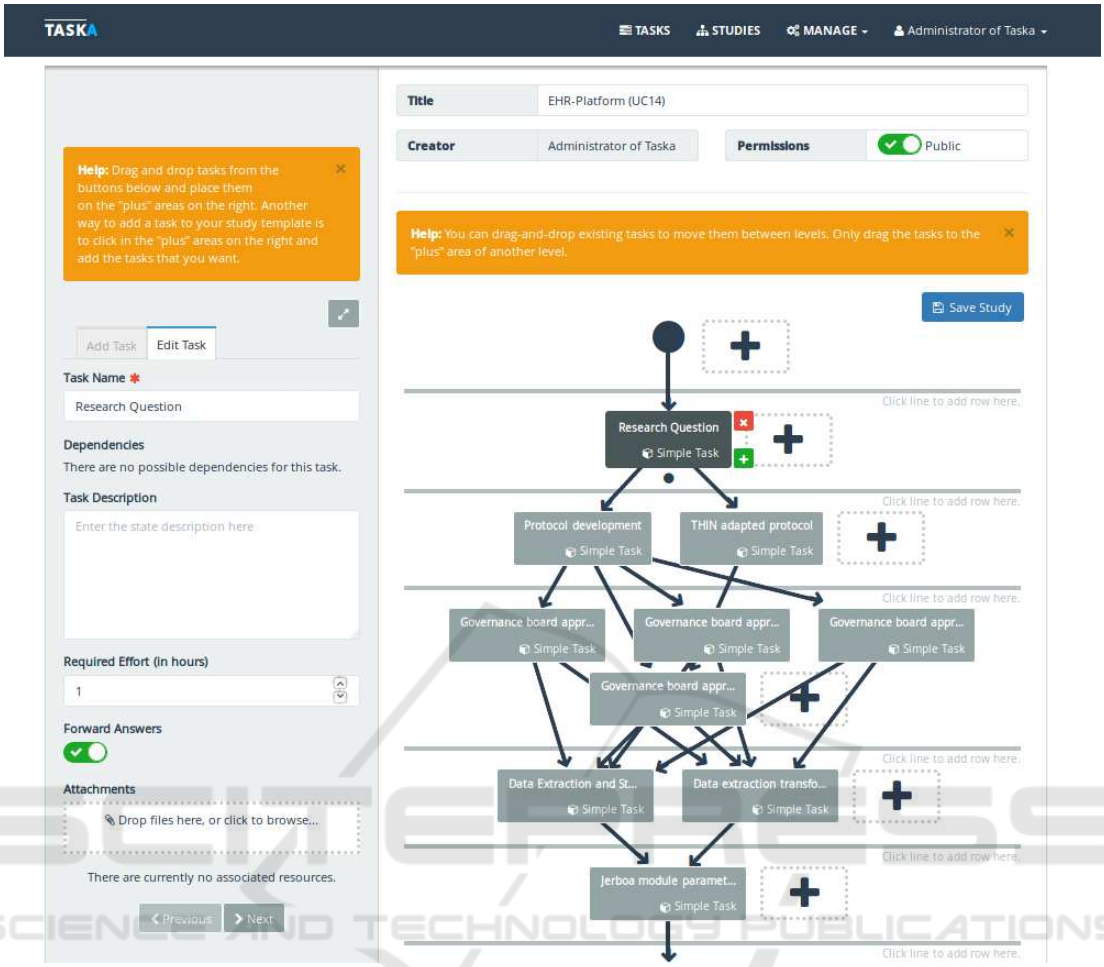


Figure 3: Study template - edit view.

the central element is a task. The combination of ReactJS and Reflux generates a three-tier structure composed of Actions, Store and View, which is fundamental for all the applications. To allow communication to the backend, all the main components of our client applications have these three sub-components.

### 4.3 Deployment

Building software solutions based on multiple libraries can make deployment very complex. Although it is not a development problem, optimizing deployment is one of the biggest challenges from a system manager perspective. To simplify this job, we decided to virtualise the application in containers, using the Docker[17] technology. This software container platform introduces just a small overhead in the host machine, allowing more applications in the same machine, or even splitting them across several containers. With this kind of deployment, it is very easy to get a running instance in a few minutes, for any operating system (Figure 5).

## 5 DISCUSSION

The user interface is undoubtedly the most important feature of any application. Users generally evaluate software's usability and functionality by briefly exploring the user interface. Our system went through several changes over time so that it could be a simple, dynamic and useful software.

To facilitate navigation we associated each concept model with a proper workspace, e.g. Tasks and Studies, and adopt familiar interface metaphors, such as the one used in a common mail reader. Figure 3 presents an example of one workflow.

The task/workflow management solution pre-

[17] https://www.docker.com

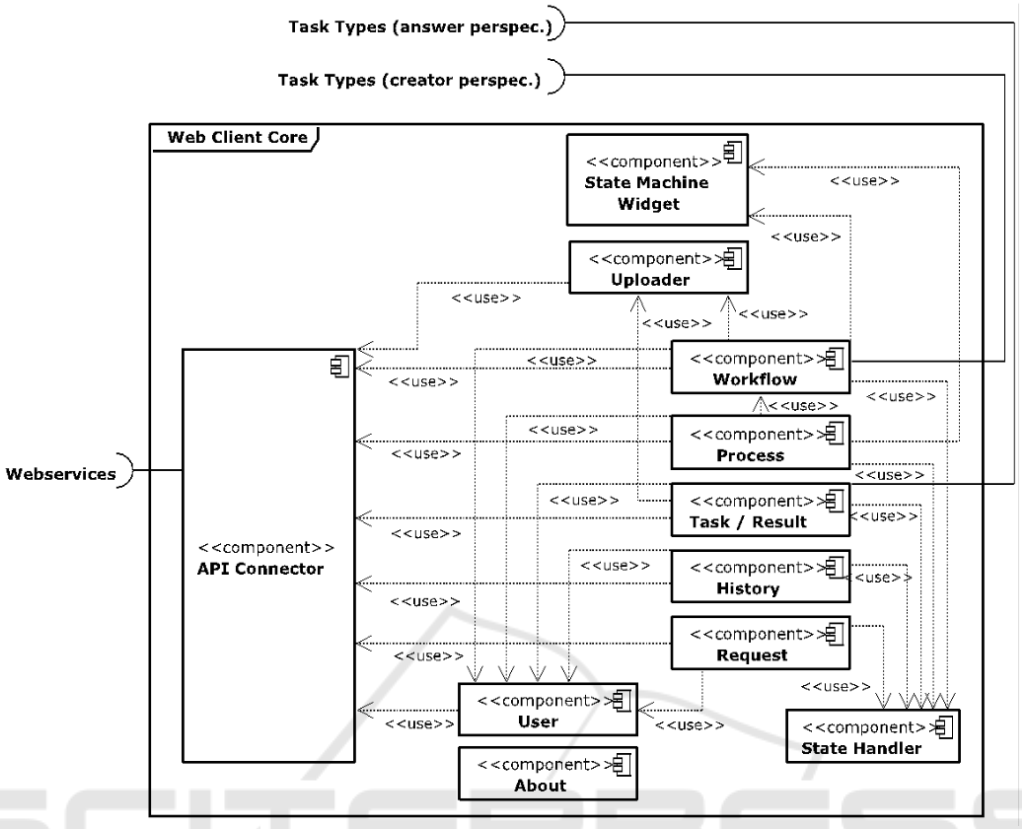


Figure 4: Client-side Architecture.

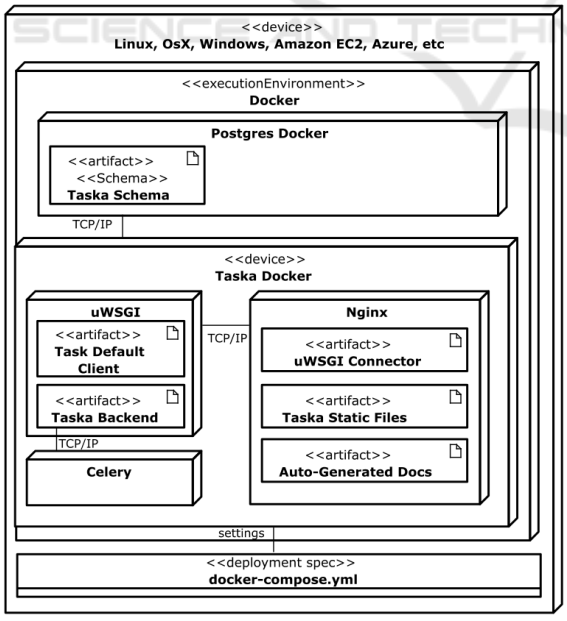


Figure 5: Deployment organization.

sented is currently being used in the EMIF project to manage research study over European patient registers, for cohorts and observation data (Vaudano et al., 2015). From this experience, we will analyse the final result aiming for the best user interface and broader use.

## 6 CONCLUSIONS

The secondary use of health records, observational and disease specific, has been the goal of many research initiatives. However, the procedures that mediate between the initial research question and the final result are still very complex and time-consuming, and normally take several months or even years to be completed. To address this scenario, we developed a task/workflow management system aiming to simplify and speed up these processes.

The platform developed fulfils a set of predefined requirements and its hybrid approach, of an object-documented system with a state machine structure, opens the door to new fields besides health studies management. Its decoupled architecture, with REST web services, allows the core system to be reused in different applications and for distinct goals.

Currently, it is a platform in development where

there are functionalities to be implemented, such as the existence of other types of tasks and the possibility of existing user groups. The creation of a new type of task is due to the need, in some situations, to repeat a task several times, producing in this way different outputs. In order to avoid this complexity in the workflow, and because the system is prepared to create new types of tasks, the next step will be to implement a task that can be completed several times. The possibility of groups of users is a necessity that is beginning to be felt due to the growth of registered users, and with the existence of groups, the creation of studies becomes faster, because the study manager can simply create a group and reuse it whenever necessary.

## ACKNOWLEDGEMENTS

This work has received support from the EU/EFPIA Innovative Medicines Initiative Joint Undertaking (EMIF grant n. 115372)

## REFERENCES

Afgan, E., Baker, D., Van den Beek, M., Blankenberg, D., Bouvier, D., Čech, M., Chilton, J., Clements, D., Coraor, N., Eberhard, C., et al. (2016). The galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2016 update. *Nucleic acids research*, 44(W1):W3–W10.

Bastião, S. L., Días, C., van der Lei, J., and Oliveira, J. L. (2015). Architecture to summarize patient-level data across borders and countries. *Studies in health technology and informatics*, 216:687–690.

Burgun, A., Bernal-Delgado, E., Kuchinke, W., van Staa, T., Cunningham, J., Lettieri, E., Mazzali, C., Oksen, D., Estupiñan, F., Barone, A., et al. (2017). Health data for public health: Towards new ways of combining data sources to support research efforts in europe. *Yearbook of Medical Informatics*, 26(01):235–240.

Ferraiolo, D., Cugini, J., and Kuhn, D. R. (1995). Role-based access control (rbac): Features and motivations. In *Proceedings of 11th annual computer security application conference*, pages 241–48.

Georgakopoulos, D., Hornick, M., and Sheth, A. (1995). An overview of workflow management: From process modeling to workflow automation infrastructure. *Distributed and parallel Databases*, 3(2):119–153.

Gini, R., Schuemie, M., Brown, J., Ryan, P., Vacchi, E., Coppola, M., Cazzola, W., Coloma, P., Berni, R., Diallo, G., et al. (2016). Data extraction and management in networks of observational health care databases for scientific research: a comparison of eu-adr, omop, mini-sentinel and matrice strategies. *eGEMs*, 4(1).

Liu, J., Pacitti, E., Valduriez, P., and Mattoso, M. (2015). A survey of data-intensive scientific workflow management. *Journal of Grid Computing*, 13(4):457–493.

Lopes, P., Silva, L. B., and Oliveira, J. L. (2015). Challenges and opportunities for exploring patient-level data. *BioMed research international*, 2015.

Oliveira, J. L., Lopes, P., Nunes, T., Campos, D., Boyer, S., Ahlberg, E., Mulligen, E. M., Kors, J. A., Singh, B., Furlong, L. I., et al. (2013). The eu-adr web platform: delivering advanced pharmacovigilance tools. *Pharmacoepidemiology and drug safety*, 22(5):459–467.

Papazoglou, M. P. (2003). Service-oriented computing: Concepts, characteristics and directions. In *Proceedings of the Fourth International Conference on Web Information Systems Engineering, WISE 2003*, pages 3–12. IEEE.

Richards, M. (2015). *Software architecture patterns*. O'Reilly Media, Incorporated.

Sheng, Q. Z., Qiao, X., Vasilakos, A. V., Szabo, C., Bourne, S., and Xu, X. (2014). Web services composition: A decades overview. *Information Sciences*, 280:218–238.

Vaudano, E., Vannieuwenhuyse, B., Van Der Geyten, S., van der Lei, J., Visser, P. J., Streffer, J., Ritchie, C., McHale, D., Lovestone, S., Hofmann-Apitius, M., et al. (2015). Boosting translational research on alzheimer's disease in europe: The innovative medicine initiative ad research platform. *Alzheimer's & dementia: the journal of the Alzheimer's Association*, 11(9):1121–1122.

vom Brocke, J., Zelt, S., and Schmiedel, T. (2016). On the role of context in business process management. *International Journal of Information Management*, 36(3):486–495.

Wolstencroft, K., Haines, R., Fellows, D., Williams, A., Withers, D., Owen, S., Soiland-Reyes, S., Dunlop, I., Nenadic, A., Fisher, P., et al. (2013). The taverna workflow suite: designing and executing workflows of web services on the desktop, web or in the cloud. *Nucleic acids research*, 41(W1):W557–W561.