YAWL in the Cloud: Supporting Process Sharing and Variability

D.M.M. Schunselaar*, H.M.W. Verbeek*, H.A. Reijers*, and W.M.P. van der Aalst*

Eindhoven University of Technology,
P.O. Box 513, 5600 MB, Eindhoven, The Netherlands
{d.m.m.schunselaar,h.m.w.verbeek,h.a.reijers,w.m.p.v.d.aalst}@tue.nl

Summary. The cloud is at the centre of attention in various fields, including that of BPM. However, all BPM systems in the cloud seem to be nothing more than an installation in the cloud with a web-interface for a single organisation, while cloud technology offers an excellent platform for cooperation on an intra- and inter-organisational level. In this paper, we show how cloud technology can be used for supporting different variants of the same process (due to "couleur locale"), and how these organisations can aid each other in achieving the completion of a running case. In this paper we describe how we have brought a BPM system (YAWL) into the cloud that supports variants[2].

Key words: BPM, Cloud, YAWL, Process Variability, Process Cooperation, Configurable Process Models

1 Introduction

In the CoSeLoG project*, 10 Dutch municipalities collaborate to see how cloud technology can be used to share resources and exchange knowledge. Of course, by bringing these municipalities into the cloud, we gain well-accepted benefits associated with the cloud. First, instead of having to buy and administer their own servers, the municipalities can simply use the cloud and focus more on their core processes. The municipalities still have to administer their own processes, but at least they do not have to administer the hardware these processes are running on. Second, by using the cloud, they can dynamically scale the required hardware up or down, depending on the current need. For example, if due to a change of legislation getting a building permit will become more difficult, then one can expect a rise in the number of applications for a building permit before the new building permit legislation comes into place. One can scale up the amount of hardware required before the bulk of applications arrives. When the volume

* This research has been carried out as part of the Configurable Services for Local Governments (CoSeLoG) project (http://www.win.tue.nl/coselog/).

[2] The installation manual and files can be downloaded from http://www.win.tue.nl/coselog/wiki/yawlinthecloud

of applications drops, one can scale down again. Furthermore, during peaks, a municipality can be aided by staff of another municipality. As a result, using the cloud will be cheaper and more flexible for a municipality, in a way that is similar to the advantages that the cloud brings to other organisations.

What sets the municipalities apart from many organisations, though, is that they are not competing with each other. For example, a citizen of Eindhoven cannot go to the Amsterdam municipality to apply for a building permit. Clearly, this citizen has no other choice than to come the Eindhoven municipality to apply for this building permit. As such, every municipality has its own, exclusive, collection of customers.

This presents us with a setting which is quite different from a regular cloud setting. As the municipalities are not competing with each other, they are quite willing to exchange ideas, to learn from each other, and to share insights with each other. As a result, when bringing municipalities into the cloud, there is no need to assume that they, as cloud tenants, are to be kept strictly separated. As a result of this, in the CoSeLoG project, we can model the similar processes of different municipalities using a single *configurable process model*. All municipalities share this single configurable process model, but they all have configured the model at deploy-time to cater for their own "couleur locale". Using such a configurable process model, one can capture common behaviour while still leaving room for some differences. As a result, all municipalities use a process model that is based on the same model (the configurable process model), but they all may run different process models.

In this paper, we assume that the cloud tenants are using variants of some super process model, are willing to cooperate, and are willing to disclose information about their processes. How can cloud technology then be best used to their advantage? This paper will show that advantages include *deploy-time* advantages, *run-time* advantages, and *post-run-time* advantages. To showcase the feasibility of a cloud implementation, we provide a proof-of-concept implementation that supports configurable process models and demonstrates these advantages.

The remainder of this paper is organised as follows: Sect. 2 explains configurable process models and lists the deploy-time, run-time, and post-run-time advantages. In Sect. 3, we present our proof-of-concept implementation. The proof-of-concept implementation is showcased by means of a scenario in Sect. 4. Relevant related work is discussed in Sect. 5. Finally, the conclusions are presented in Sect. 6.

2 Advantages of Cloud Technology

By combining configurable processes and cloud technology, we can achieve advantages in three areas: deploy-time, run-time, and post-run-time. Prior to going into these advantages, we first explain configurable process models as these are on the basis of some of the advantages.

Configurable Process Models Configurable process models describe a family of process models using variation points. Variation points are locations in the process model which can be modified by the user. In other words, the municipality can select to remove that part from the model or substitute that part of the model with a model fragment from

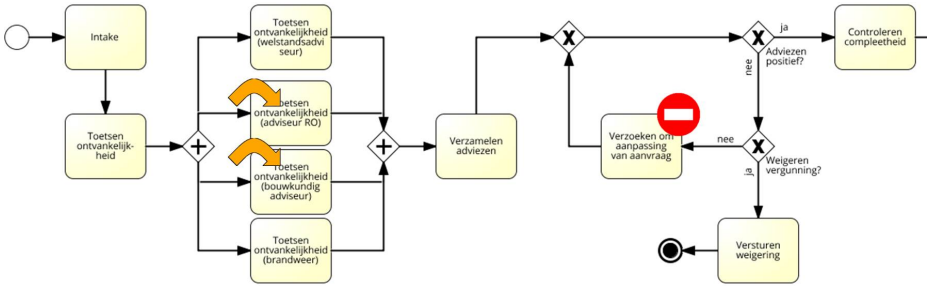

Fig. 1: Part of the configurable process model from the scenario.

a predefined set of model fragments. When the user has set all the variation points, we have a process model that can be executed by a BPM system.

In the configurable process model (Fig. 1), we have the option to *hide* certain parts (orange curved arrow) and we have the option to *block* certain parts (no-entry sign). When a certain part is hidden, that part is substituted by an automatic task. Blocking a certain part results in the removal of that part from the model. Taking Fig. 1 as an example, hiding "Toetsenontvangekijkheid (adviseur RO)" entails replacing that activity with an automatic task. Blocking "Verzoeken om aanpassing van aanvraag" means that the exclusive choice "Weigeren vergunning" will always evaluate to "Versturen weigering". Note that activities do not vary from model to model, i.e., the presence of an activity can be changed but not the contents of the activity. In case of the configurable process model in Fig. 1, the activity "Toetsenontvangekijkheid (adviseur RO)" can be hidden in a model but if it is not hidden then it is the same as "Toetsenontvangekijkheid (adviseur RO)" in any other obtainable model.

Deploy-time Advantages A municipality that wants to deploy a process model in this cloud setting does not have to create a model from scratch. Instead, it only needs to configure an existing model from a configurable process model.

In the classical setting, each municipality maintains its own process models and IT infrastructure. This means that every change in legislation has to be incorporated by every municipality. When moving to the cloud using configurable process models, changes mainly have to be incorporated in the configurable process model. Municipalities might have to change their configuration. In Fig. 2, the old situation is compared to a hypothetical cloud situation. Although the maintenance efforts for the configurable process model are larger than for the individual models (it is more complex), the total maintenance effort is smaller than the sum of maintenance efforts of each of the municipalities.

Run-time Advantages Next to the deploy-time advantages, the municipalities can also expect run-time advantages. Some of the run-time advantages are directly related to the use of the cloud, i.e., scalability, availability, reliability, cost reduction, etc. Other advantages during run-time, however, amount to an increase in the flexibility and robustness of the organisation.

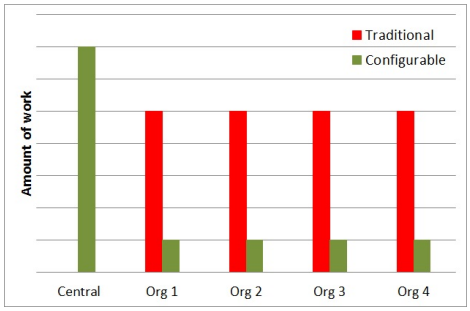

Fig. 2: The expected benefit in maintenance when municipalities move to the cloud using a configurable process model.

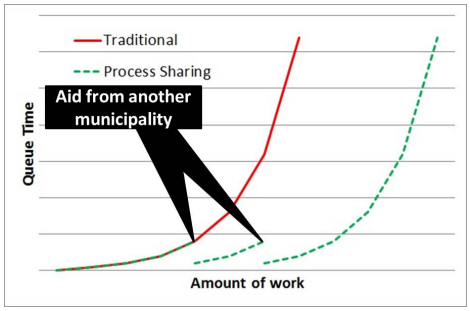

Fig. 3: In the traditional situation, the queue time increases significantly when the amount of work increases. Using Process Sharing, we expect that employees from other municipalities can aid to postpone the significant increase in queue time.

The expected increase in flexibility is achieved by the fact that municipalities are capable of allocating work to resources from another municipality. In the traditional situation (Fig. 3), we have the well-known curve related to the PASTA property, i.e., when the amount of work increases this results in the utilisation rate approaching 1 yielding a queue time which goes to infinity. By sharing the execution of the process between municipalities, other municipalities can offer staff when the queue time becomes too long (until there are no resources left). Next to this, municipalities can also share an expert to aid them in their executions. The addition of an expert means that part of the execution of a case is partly outsourced. This flexibility has as added advantage that the robustness increases. For instance, in case of disasters (flooding, power failure, etc.), staff of other municipalities can aid. One can argue that the process models are different between municipalities and thus aiding another municipality requires learning the other's process model. However, as mentioned with the configurable process model, the process models might be different but the individual activities do not differ with respect to content. This means that if a municipality also executes a particular activity, then that municipality can aid in executing that activity.

Fig. 4: In the traditional situation, a municipality can only compare itself to itself. With the use of a single cloud service and a configurable process model, we expect a graph similar to the right one where municipalities can benchmark.

Post-run-time Advantages Traditionally, municipalities are only able to look at themselves. By having comparable executions, municipalities can benchmark themselves with respect to others (see Fig. 4 for a hypothetical graph). The comparability of the executions comes forth from the fact that all models are deduced from a configurable process model and the use of a single cloud service which guarantees uniform naming conventions, same level or granularity, and comparable data structures.

3 Proof-of-concept Implementation

To be able to show the advantages sketched earlier, we made a proof-of-concept implementation. For our proof-of-concept implementation we have chosen YAWL as our BPM system since YAWL has the following advantages: components are decoupled making them ideal to be run in distributed mode, and native support for configurable process models [1]. For the cloud provider, we have chosen Microsoft Azure. Finally, YAWL in the cloud runs on the PaaS (Platform as a Service) layer.

We managed to make YAWL available in the Azure cloud without making changes to YAWL itself. This has the advantage that updates of YAWL can be used and there is no need to maintain a special YAWL version. Furthermore, by not changing YAWL itself, we maintained the look and feel people are familiar with; removing the need to learn a new system. Finally, we created a component to control the cloud allowing administrators to, amongst others, upload configurable process models.

YAWL [1] is a workflow system based on Petri net semantics. YAWL has an architecture which is service-oriented. Part of the architecture of YAWL is depicted in Fig. 5. The individual components, e.g., the engine, resource service, etc., are independent components which are coupled using different interfaces. In this paper, we "cloudify" the engine, therefore, we briefly touch upon the interface A, B, E, and X (highlighted in the red rectangle in Fig. 5). Interface A is used for, amongst others, loading and unloading process specifications. Interface B is used for most of the handling of work items and creating new process instances. Interface E is used for the retrieval of process logs. Finally, interface X is used for exception handling.

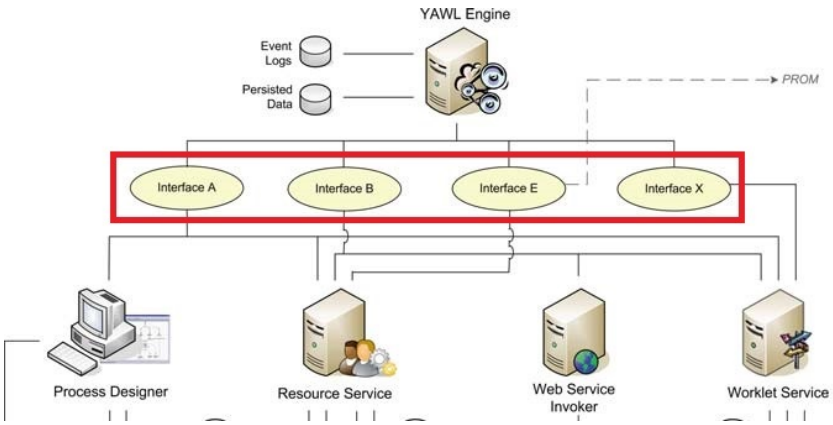

Fig. 5: Part of the architecture of YAWL. Figure taken from [1].

As mentioned, YAWL consists of components which communicate with each other using different interfaces. In a non-cloud based YAWL installation, there is a single engine. This engine receives requests on its interfaces and acts accordingly. With bringing YAWL into the cloud, we want to allow for multiple engines running concurrently. In order to be able to scale up and down, we want to use a dynamic amount of engines. Furthermore, the other components within YAWL expect to communicate with a single engine. Therefore, within YAWL in the cloud, we have created an abstraction from the engines allowing for multiple engines to be used and at the same time offer a single set of interfaces (A, B, E, and X) to the outside world to communicate with. This results in the high-level architecture shown in Fig. 6.

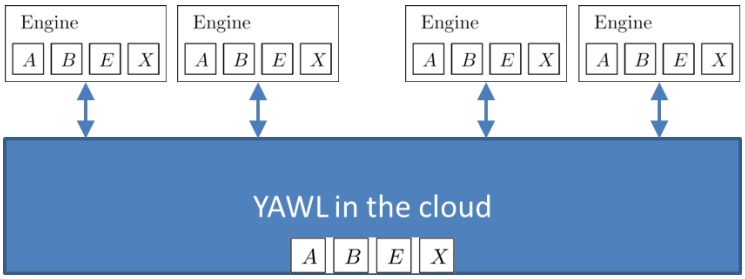

Fig. 6: The high-level architecture for YAWL in the Cloud.

By using this architecture, we do not need to make any changes to YAWL. Furthermore, by offering the same set of interfaces to the outside world, there is no change noticeable, i.e., one cannot see the difference between a single engine or the entire cloud. However, since we do not make any changes to YAWL, the YAWL engines are oblivious of each other. This means that, for instance, the case identifiers are unique per engine, but not amongst engines. Furthermore, engines might now be used for multiple

organisations resulting in cases running in different contexts for a single engine (engines normally run within a single context). To accommodate for this, we propose the more detailed architecture shown in Fig. 7.

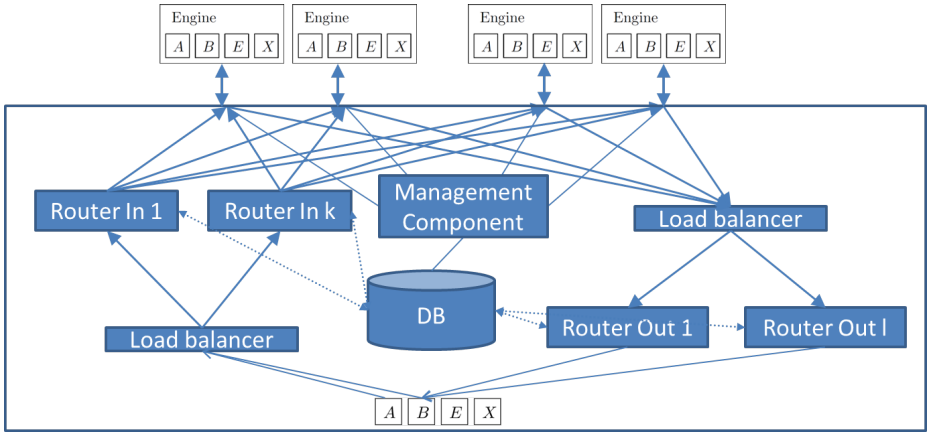

Fig. 7: The detailed architecture for YAWL in the Cloud.

As shown in Fig. 7, there is a central cloud based database which is used to transform case identifiers etc. from the local context of an engine, to the global context of the cloud and vice versa. By using a cloud based database, all scalability challenges are handled by the cloud. Furthermore, we have introduced routers to route the various requests to the correct engine(s), e.g., when an organisation wants to know all the cases currently running for that organisation, the router performs a lookup to see which engines to contact. After contacting the various engines, the router combines each of their responses into a single response as this is expected by the environment. Finally, since we have distributed the engines, we also want to distribute the routers for the same reasons, therefore, we can use a cloud based load balancer to automatically forward requests to the least busy router. This cloud based load balancer scales automatically up and down without any involvement.

In the centre of the detailed architecture, we have a management component. This management component can query the database for information on the various engines. Furthermore, it can be used to add/remove available engines; the enablement and disablement of engines is handled by the cloud.

Most notably for the implementation is the fact that YAWL in the cloud, similar to YAWL, is implemented in java. Furthermore, we use Hibernate[3] as abstraction layer from the database. Both java and Hibernate make YAWL in the cloud largely platform independent. Unfortunately, YAWL did not work properly with the cloud based version of MSSQL, therefore, we have used a virtual machine with MySQL. The implementation and an installation manual can be downloaded from `http://www.win.tue.nl/coselog/wiki/yawlinthecloud`.

[3] `http://hibernate.org`

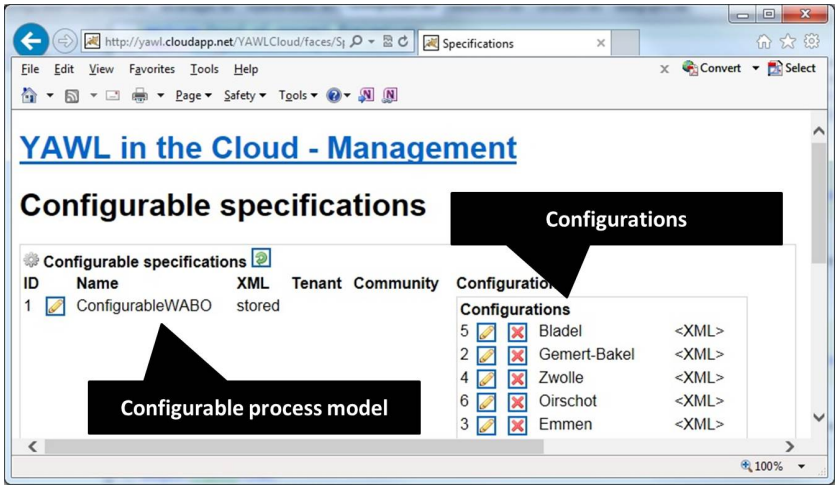

Fig. 8: A configurable process model in the cloud with the configurations for the various municipalities.

4 Proof-of-concept Scenario

We evaluate our implementation by means of a hypothetical scenario. In this hypothetical scenario, all the CoSeLoG municipalities want to cooperate with each other in the cloud. To reap the deploy-time benefits, they obtain a configurable process model capable of supporting their processes (part of this model is depicted in Fig. 1). Next to the configurable process model, the municipalities use the Synergia toolset [2] to define their configurations. The configurable process model and configurations are uploaded to the cloud (Fig. 8).

Using the Synergia toolset, the configuration can be projected on the configurable process model to obtain an executable process model. These are added to the set of loadable specifications (Fig. 9).

To give the municipalities the possibility to work on their cases, virtual machines are created in Microsoft's Azure cloud (Fig. 10 contain the virtual machines for Emmen and Gemert-Bakel). Each municipality is offered a slightly customised portal to YAWL in the cloud where already some of their cases are present (Fig. 11).

Assume all employees able to handle the process in Gemert-Bakel get ill. Luckily, we have the run-time benefits of the cloud and promptly employees from Emmen are logging in to aid Gemert-Bakel in the execution of their processes (Fig. 12).

A similar scenario like this has been presented to the participating municipalities of the CoSeLoG project in our yearly meeting. The contact persons were subdivided into municipalities and got some hands-on experience with YAWL in the cloud. The various contact persons were enthusiastic about the presented implementation. However, this

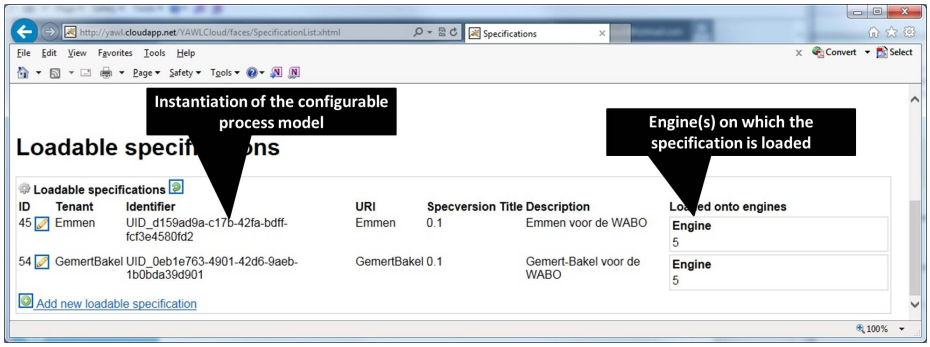

Fig. 9: Specifications for some municipalities have been uploaded to a (shared) engine.

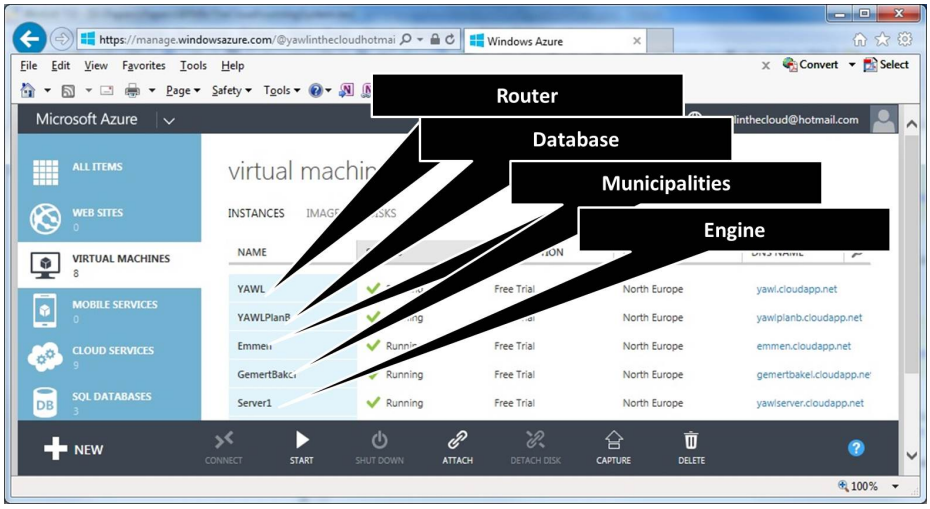

Fig. 10: The various virtual machines for the municipalities, a router, an engine, and a database server.

was not a real evaluation but more a small showcase to show the work conducted in the project.

5 Related Work

Both in academia and industry, there has been a lot of interest in BPM/WFM in the cloud.

Academic publications In [3], the authors bring BPEL to the cloud. The authors extensively discuss different considerations for bringing BPEL to the cloud using different

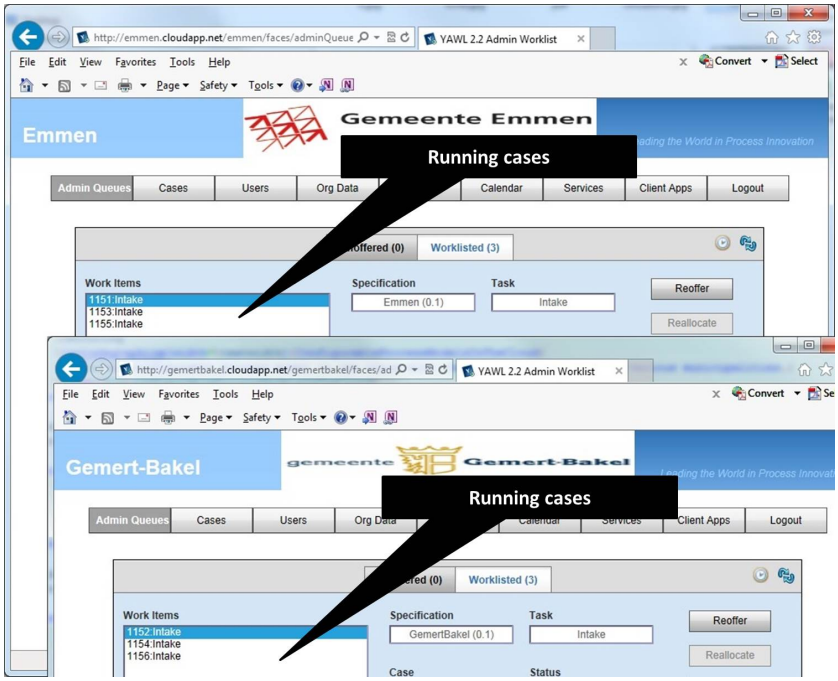

Fig. 11: The portal for Emmen and Gemert-Bakel with some running cases.

levels, i.e., infrastructure, platform, and software, and security considerations. The authors state that they are busy with modifying an open-source BPEL engine to be used in the cloud. In [4], the authors add configurability to BPEL in an extension called *VxBPEL*. In [5], configurable BPEL is presented. However, for both approaches there is no graphical editor making it cumbersome to maintain the models.

In [6], ARIS in the cloud is presented where resources can be shared amongst different locations around the world. It is unclear whether this is based on a single process model being used for all the branches. Although this approach is not directly applicable to the municipality setting, the approach can be beneficial for companies with multiple branches, e.g, Hertz.

The workflow engine CPEE[4] [7] offers a cloud based workflow engine. This workflow engine has been built from scratch and allows for run-time modifications of the process model. It is designed with a single organisation in mind.

Industrial solutions As mentioned, there exist a multitude of cloud solutions from the industry. However, most of these solutions seem nothing more than a web-based interface to a classical BPM system running on the servers of the vendor or outsourced to a third party. We list the cloud based BPM systems based on Gartners Magic Quadrant for Intelligent Business Process Management Suites [8]. We do not list all of the com-

[4] `www.cpee.org`

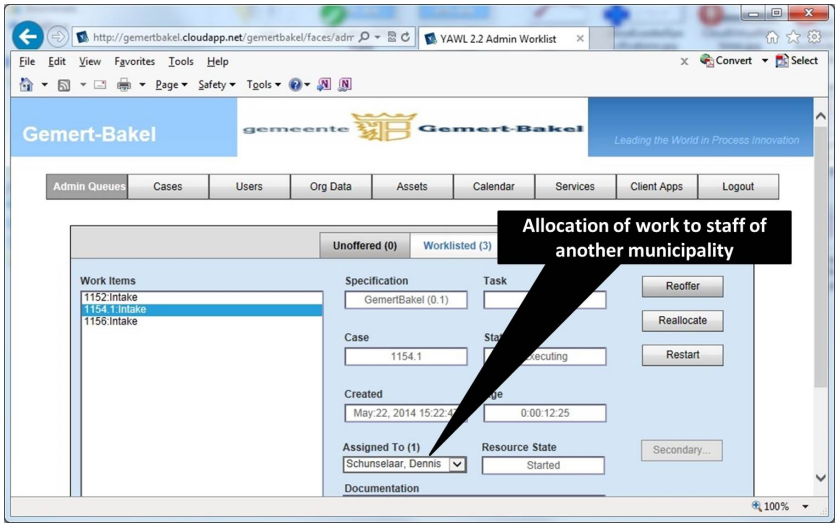

Fig. 12: An employee from Emmen (Dennis Schunselaar) is executing cases for Gemert-Bakel.

panies but mention only the ones where there is a strong cloud platform according to Gartner.

Kofax [9] offers a cloud platform called TotalAgility. One of the features of Kofax is the use of "process skins". Process skins allow the user to manage multiple versions of the same process type. Whenever there is an update to the process all skins are updated accordingly. This seems to be similar to configurable process models, but the expressive power and capabilities are not specified. Finally, TotalAgility reasons with a single organisation in mind.

Other solutions mentioned do not support configurable process models, these include: Appian [10], OpenText [11], PNMSoft [12], and Software AG [13].

6 Conclusion

We have sketched the added benefits of bringing non-competing organisations like municipalities, court houses, ministries, etc. into the cloud. These benefits include well-known cloud benefits like: scalability, availability, cost reduction etc. But these benefits can be extended towards deploy-time advantages (by using configurable process models), run-time advantages (increase in flexibility and robustness), and post-run-time advantages (benchmarking with other municipalities).

Next to sketching the benefits, we have provided an implementation where we brought YAWL into the cloud. Within YAWL in the cloud, we support multiple organisations, and we offer the possibility to support multiple variants of the same process using configurable process models.

To show process variability is possible in the cloud, we presented a proof-of-concept implementation of YAWL in the cloud. In this proof of concept implementation, we show we did not need to change YAWL. Furthermore, in the evaluation, we showed the cloud infrastructure is invisible. Finally, we showed parts of the component capable of controlling the cloud.

In this paper, we focussed on the setting of non-competitive organisations working together, specifically municipalities. Also, note that this approach can be beneficial within a single organisation as well. Take for instance Hertz, which has numerous branches in numerous countries. Instead of having an installation per branch, there can now be a centralised BPM system in the cloud in which the various branches can cooperate.

Acknowledgements

The authors would like to thank T.F. van der Avoort for his work on the implementation of YAWL in the cloud as part of his master thesis [14].

References

1. Hofstede, A.H.M. ter, Aalst, W.M.P. van der, Adams, M., Russell, N., eds.: Modern Business Process Automation: YAWL and its Support Environment. Springer (2010)
2. La Rosa, M., Gottschalk, F.: Synergia - Comprehensive Tool Support for Configurable Process Models. In de Medeiros, A.K.A., Weber, B., eds.: BPM (Demos). Volume 489 of CEUR Workshop Proceedings., CEUR-WS.org (2009)
3. Anstett, T., Leymann, F., Mietzner, R., Strauch, S.: Towards BPEL in the Cloud: Exploiting Different Delivery Models for the Execution of Business Processes. In: SERVICES I, IEEE Computer Society (2009) 670–677
4. Koning, M., ai Sun, C., Sinnema, M., Avgeriou, P.: VxBPEL: Supporting variability for Web services in BPEL. Information & Software Technology **51**(2) (2009) 258–269
5. Gottschalk, F.: Configurable Process Models. PhD thesis, Eindhoven University of Technology, The Netherlands (2009)
6. Scheer, A.W., Klueckmann, J.: BPM 3.0. In Dayal, U., Eder, J., Koehler, J., Reijers, H.A., eds.: BPM. Volume 5701 of Lecture Notes in Computer Science., Springer (2009) 15–27
7. Stuermer, G., Mangler, J., Schikuta, E.: Building a modular service oriented workflow engine. In: SOCA, IEEE (2009) 1–4
8. Jones, T., Schulte, W.R., Contara, M.: Magic Quadrant for Intelligent Business Process Management Suites. Gartner **G00255421** (2014)
9. Kofax: TotalAgility. http://www.kofax.com/smart-process-application-platform/ Last accessed 04-04-14.
10. Appian: Appian. http://www.appian.com/bpm-software/cloud-bpm.jsp Last accessed 04-04-14.
11. OpenText: OpenText Cordys. http://www.opentext.com/What-We-Do/Products/Business-Process-Management/Process-Suite-Platform/BPM-in-the-Cloud Last accessed 04-04-14.
12. PNMSoft: Cloudworks. http://www.pnmsoft.com/technology/cloud-bpm/ Last accessed 04-04-14.
13. SoftwareAG: webMethods BPMS. http://www.softwareag.com/corporate/products/wm/bpm/products/bpms/overview/default.asp Last accessed 04-04-14.
14. Avoort, T. F. van der: BPM in the Cloud. Master's thesis, Eindhoven University of Technology, The Netherlands (2013)