SoftwareX 12 (2020) 100576

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

YAWL: An open source Business Process Management System from science for science

Michael Adams [a,*], Andreas V. Hense [b], Arthur H.M. ter Hofstede [a]

[a] Queensland University of Technology, Brisbane, Australia
[b] Bonn-Rhein-Sieg University oAS, Sankt Augustin, Germany

ARTICLE INFO

Article history:
Received 4 March 2020
Received in revised form 29 July 2020
Accepted 30 July 2020

Keywords:
BPMS
Open source software
YAWL
Workflow
process

ABSTRACT

YAWL (*Yet Another Workflow Language*) is an open source Business Process Management System, first released in 2003. YAWL grew out of a university research environment to become a unique system that has been deployed worldwide as a laboratory environment for research in Business Process Management and as a productive system in other scientific domains.

Code metadata

Current code version	v4.3.1
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX_2020_80
Code Ocean compute capsule	None
Legal Code License	LGPL-3.0
Code versioning system used	git
Software code languages, tools, and services used	Java, JavaScript
Compilation requirements, operating environments & dependencies	Apache ant; Linux, Windows, OSX; Java 8 or later
Link to developer documentation/manual	https://yawlfoundation.github.io/assets/files/YAWLTechnicalManual4.pdf
Support email for questions	yawl.michael@gmail.com

Software metadata

Current software version	v4.3.1
Permanent link to executables of this version	https://github.com/yawlfoundation/yawl/releases/tag/v4.3.1
Legal Software License	LGPL-3.0
Computing platforms/Operating Systems	Linux, Windows, OSX
Installation requirements & dependencies	Java 8 or later
Link to user documentation/manual	https://yawlfoundation.github.io/assets/files/YAWLUserManual4.3.pdf
Support email for questions	yawl.michael@gmail.com

1. Motivation and significance

A business process is a collection of identifiable tasks that need to be performed in a certain order to achieve some business goal. Fig. 1 shows a graphical model of a simple example business process [1] depicting the control flow – i.e. the order in which the

* Corresponding author.
E-mail addresses: mj.adams@qut.edu.au (M. Adams), andreas.hense@h-brs.de (A.V. Hense), a.terhofstede@qut.edu.au (A.H.M. ter Hofstede).

https://doi.org/10.1016/j.softx.2020.100576

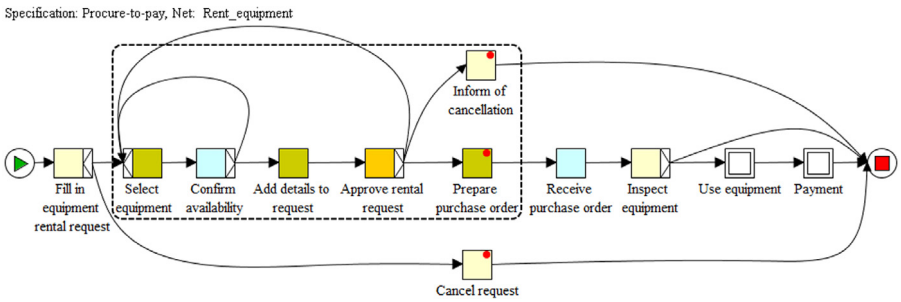

Fig. 1. An example YAWL process model.
Source: From [1].

tasks are executed – by arrows and certain operators for forking and merging. The resources – i.e. the participant roles that are to perform the tasks – are depicted by colours here. Business Process Management Systems (BPMSs) add a third perspective, namely the data that are needed to perform each task. With defined specifications encapsulating these three perspectives, a BPMS can generate an IT system to support and partially automate business processes.

At the turn of the century, there existed many commercial and open source BPMSs, each coming with its own graphical notation. There were two major efforts to standardise these notations. The first was a consortium of the major players in the field. The results were serialisation standards for business process definitions, namely XPDL 1.0 and XPDL 2.0, and a graphical language called BPMN [2].

The second endeavour undertook a scientific analysis of all patterns of constructs used in BPM systems at that time [3,4]. Deriving from that work was a minimal but highly expressive graphical language together with a formal definition of its semantics. The implementation of this language yielded yet another BPMS, called YAWL [5]. By design, YAWL should be the most powerful language for process specification. It offers comprehensive support for the vast majority of the identified control-flow, data, resource, and exception handling patterns, and thus YAWL is able to manage processes of practically any complexity [5].

Over the years, YAWL has become a full-fledged BPMS that emphasises modularity, extensibility, robustness and ease of use, and is not only a research platform for Business Process Management but can be used productively in a wide range of scientific and organisational settings.

2. Software description

YAWL is a BPMS consisting of an editor for the definition of process specifications and an engine for their execution. Process specifications contain all perspectives that are necessary for process automation: the control flow, the resources, and the data. The control flow is edited as a sequence of tasks in a graphical form as shown in Fig. 1. How resources and data for each task are specified is discussed in Section 2.2.1.

2.1. Software architecture

The YAWL environment is based on a service-oriented architecture (SOA), with a core execution engine and a set of ancillary components implemented as RESTful Web services. This modular design allows for the modification and addition of components, according to specific requirements. A high-level overview of the YAWL architecture is shown in Fig. 2. The YAWL Engine is the central component that manages the creation and execution of cases, and keeps track of where each case is in its control flow, i.e. its *current state*. When a human task is ready to be performed, the engine deploys it to the Resource Service, which then assigns a user, in accordance with the process specification. Automated tasks – for example, the sending of an e-mail or intensive data transformations – will be deployed to bespoke services. In a laboratory setting, for example, the integration of a machine that performs certain experiments can be realised by developing a dedicated support service for it.

The Administration Module allows for uploading new specifications, reassigning work items, cancelling cases, etc. The Worklist Handler shows each user the work items that are currently assigned to them. When the user opens one of these work items, the Task Data Input Forms Generator will automatically generate an appropriate web form based on the data types and values of the corresponding task of the specification.

All YAWL services are deployed within a servlet container; the standard YAWL distribution is bundled with Apache Tomcat.[1] Hibernate ORM[2] manages the persistence of the system and supports a wide range of relational DBMSs.

2.2. Software functionalities

With YAWL, processes can be designed, executed, and analysed. Because of its high modularity, every part of the environment, including all user interfaces, can be modified or replaced by users and developers.

2.2.1. YAWL process editor

As stated above, the YAWL process editor allows for the creation of process specifications covering all perspectives necessary for process automation. The YAWL editor allows for the creation of graphical process specifications in the YAWL language, such as those shown in Figs. 1 and 3. In addition to what is visible in the graphical diagram, there are data variables defined for each task, using XML Schema. Task variables can be created and mapped from global variables by simple drag and drop operations. If more complex operations are necessary, XPath and XQuery expressions may be used. Regarding the resource perspective, YAWL can categorise human and non-human resources into work roles, positions, and organisational capability based groupings. Based on these resources, YAWL can distribute work in many different ways [5]. Users and user groupings can be created directly within YAWL, and/or organisational data can be imported from LDAP or other related systems, such as Liferay Portal.

[1] https://tomcat.apache.org/.
[2] https://hibernate.org/.

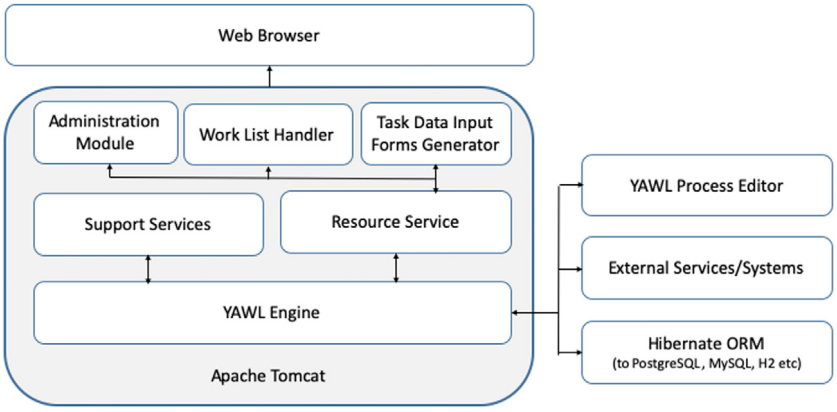

Fig. 2. A high-level architecture schematic of the YAWL environment.

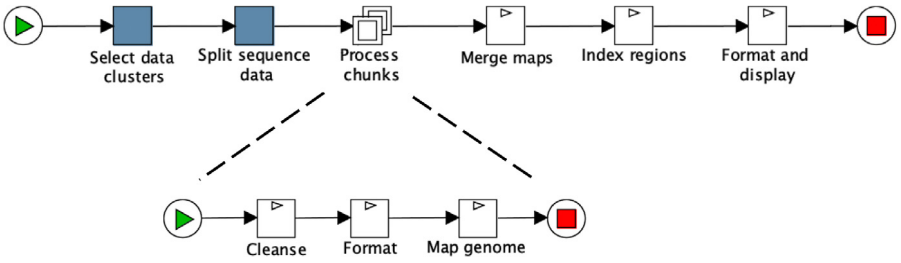

Fig. 3. The *Epigenomics* process model.

2.2.2. YAWL runtime environment

For human participants, the YAWL environment supports the automated generation of browser-based forms that display and capture data values of task input and output variables. The layout and content of the generated automated forms can be further guided with pre-set and user-defined extended attributes that may be assigned at design time and applied during the pre-rendering phase of the form at runtime. These features are particularly useful for rapid prototyping purposes, and may be fully replaced with a pluggable set of user-defined forms as needed.

2.2.3. Exception handling

YAWL offers unique support for dynamic processes and exception handling through the Worklets/Exlets approach [6]. Worklets are small process definitions that replace placeholders in a higher-level process specification at runtime, based on the context of each case, following an extensible set of rules that can be extended on-the-fly while the process is executing. Exlets are an extensible set of operational primitives and compensation processes that, when defined, will automatically execute in the event of a process error or deviation, whether or not the error was anticipated at design time [6].

2.2.4. Process logging and mining

All system and process actions are logged, and that data is made available in a variety of formats, including the open source XES format. This provides easy integration into process mining tools, such as ProM,[3] for post-execution analysis and diagnosis.

[3] http://www.processmining.org.

3. Illustrative example

The following example serves to illustrate the use of the YAWL environment to define and execute scientific processes. This example is based on the epigenomics workflow described by Juve et al. [7], which automates the generation and display of epigenetic state data from human cells. Many more thorough exemplars can be found, for example, in [5,8–12]

The process specification consists of a main (or primary) model and a sub-process model, as shown in Fig. 3. The process begins with the selection of DNA sequence data sources and the choice of a genome strand to isolate and display. The data is then split into a chosen number of *chunks*, each to be processed in parallel.

The *Process Chunks* sub-process is an example of a multiple instance sub-process, that is it is dynamically instantiated multiple times, one instance for each data chunk, depending on the number of chunks to be processed. Each sub-process instance is assigned one chunk, and begins by first cleansing it to remove noisy or contaminated segments, then converting the data to the binary format required for input into the *Map genome* task, which maps the binary DNA data into the relevant selected strand location on a reference genome.

Once all of the sub-processes have completed, each mapping is merged into a global genome map, then indexed by region. Finally, information from the specified genome region is extracted, then converted into the proper data format for display within a graphical user interface.

Like many scientific workflows, the process is quite linear, where the outputs of one task become the inputs for the next task in the sequence. The ability to succinctly define a multiple instance sub-process, which will spawn a number of instances depending on a matrix of runtime data partition values and available computing resources, effectively creates a multi-parallel,

distributed execution network, while negating the need for an over-complication of the model.

YAWL provides for the work of a task to be assigned to any resource, human or software. The first two tasks in this workflow are shaded to denote their execution by human participants. Further, they are shaded with the same colour to denote that the participant assigned the second task (e.g. a member of the *Epigenetic Investigator* role) must be the same participant who completed the first task; this is an example of the *retain familiar* resourcing pattern [4]. The remaining tasks are annotated with triangular 'play' icons to denote that those tasks are automated. In this example, each automated task is assigned to a scripting engine where bespoke routines are called to operate on the input data streams. In this way, work can be distributed across as many computing resources as are available to efficiently run those routines.

The exception handling capabilities of YAWL [6] (not depicted in Fig. 3) allow exceptions to be detected and effectively handled in real time, for example if the data cleanse routine failed to remove errant data or if there was a failure in a mapping operation, so that the process can continue unheeded. These capabilities negate the need to cancel and restart the entire process when an error occurs.

4. Impact

YAWL has been downloaded more than 250,000 times, and to over 170 countries. From prototypical beginnings, there have been more than 40 formal version updates, including three major version releases: v2, which included a new resource perspective, new administration and user interfaces, and new automated forms; v3, which incorporated a new and enhanced process editor that also supports plug-in extensions; and v4, which included a new *control panel* app that fully encapsulated the environment into a single entity. The latest version, v4.3.1, contains new security features, as well as a number of minor enhancements and updates.

The YAWL language and environment has been used extensively for research purposes. A primary factor in the applicability of YAWL for research is its extensibility: it is relatively easy to develop a new extension, service or enhancement and plug it in to the environment. Each core component of the YAWL environment has its own extensive Application Programming Interface (API). A task in a YAWL process can be delegated at runtime to any service, system, application, person or code module, which allows a task in a process model to represent any applicable action.

A *Google Scholar* search for "yawl business process management" yielded approximately 6000 academic papers citing YAWL in some way. For instance, YAWL has been used in research efforts involving: the study of process language transformations [13,14]; the definition of formal semantics of other languages using YAWL formalisations [15]; approaches to process model analysis and verification [11,16]; process flexibility extensions and techniques [8,17]; exception handling techniques during process execution [6]; process configuration methods [10]; case studies of organisational processes using YAWL [9,12,18]; and process simulation research [19], to select but a few.

While the processes in BPMSs such as YAWL are driven by the control flow (as illustrated by the process definitions in Figs. 1 and 3), in e-Research there are processes or workflows that are purely driven by the availability of data. Popular scientific workflow systems in e-Research include Taverna, Kepler, and others. Nevertheless, a project by INRIA involving high-performance computing chose YAWL for process support due to its dynamic exception handling capabilities [20].

5. Conclusion

The YAWL system continues to evolve as ideas for new uses and applications are realised. YAWL in the Cloud is one recent project where we are exploring ways to deploy a load-balanced array of YAWL engines in a cloud environment in an effort to provide BPMS services and benefits to a number of organisations, negating the need for those organisations having to deploy their own BPMS locally [21].

Blockchain integration is another developing direction, where distributed ledger technologies can be leveraged to support inter-organisational workflow without the need for a trusted intermediary [22].

Recent efforts and emerging pathways are indicative of the ongoing applicability of the YAWL environment. While there have been many benefits realised and challenges met since its first release, the YAWL system has maintained its relevance in many varied research and learning domains.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Hense Andreas V, Malz Robert. Comparison of the subject-oriented and the Petri net based approach for business process automation. In: Ehlers Jens, Thalheim Bernhard, editors. Proceedings of the 7th international conference on subject-oriented business process management, S-BPM ONE 2015. ACM; 2015, p. 21:1–21:12.
[2] Wohed Petia, van der Aalst Wil MP, Dumas Marlon, ter Hofstede Arthur HM, Russell Nick. On the suitability of BPMN for business process modelling. In: International conference on business process management. Springer; 2006, p. 161–76.
[3] van der Aalst WMP, ter Hofstede AHM, Kiepuszewski B, Barros AP. Workflow patterns. Distrib Parallel Databases 2003;14(3):5–51.
[4] Russell Nick, van der Aalst Wil MP, ter Hofstede Arthur HM. Workflow patterns: The definitive guide. Cambridge, MA: The MIT Press; 2016.
[5] ter Hofstede Arthur HM, van der Aalst Wil MP, Adams Michael, Russell Nick. Modern business process automation: YAWL and its support environment. Springer; 2009.
[6] Adams Michael, ter Hofstede Arthur HM, van der Aalst Wil MP, Edmond David. Dynamic, extensible and context-aware exception handling for workflows. In: Meersman Robert, Tari Zahir, editors. On the move to meaningful internet systems 2007: CoopIS, DOA, ODBASE, GADA, and IS. Lecture notes in computer science, vol. 4803, Springer Berlin Heidelberg; 2007, p. 95–112.
[7] Juve Gideon, Chervenak Ann, Deelman Ewa, Bharathi Shishir, Mehta Gaurang, Vahi Karan. Characterizing and profiling scientific workflows. Future Gener Comput Syst 2013;29(3):682–92.
[8] van der Aalst Wil MP, Adams Michael, ter Hofstede Arthur HM, Pesic Maja, Schonenberg Helen. Flexibility as a service. In: International conference on database systems for advanced applications. Springer; 2009, p. 319–33.
[9] Baccarin E, Madeira Edmundo Roberto Mauro, Medeiros Claudia Bauzer, van der Aalst Wil MP. Spica's multi-party negotiation protocol: Implementation using YAWL. Int J Coop Inf Syst 2011;20(03):221–59.
[10] Gottschalk Florian, van der Aalst Wil MP, Jansen-Vullers Monique H, La Rosa Marcello. Configurable workflow models. Int J Coop Inf Syst 2008;17(02):177–221.
[11] Wynn MT, Verbeek HMW, van der Aalst WMP, ter Hofstede AHM, Edmond D. Reduction rules for YAWL workflows with cancellation regions and OR-joins. Inf Softw Technol 2009;51(6):1010–20.
[12] Yu Bo, Wijesekera Duminda. Building dialysis workflows into EMRS. Proc Technol 2013;9:985–95.
[13] Decker Gero, Dijkman Remco, Dumas Marlon, García-Bañuelos Luciano. Transforming BPMN diagrams into YAWL nets. In: International conference on business process management. Springer; 2008, p. 386–9.
[14] Mendling Jan, Moser Michael, Neumann Gustaf. Transformation of yEPC business process models to YAWL. In: Proceedings of the 2006 ACM symposium on applied computing. ACM; 2006, p. 1262–6.
[15] Ye JianHong, Sun ShiXin, Song Wen, Wen LiJie. Formal semantics of BPMN process models using YAWL. In: Second international symposium on intelligent information technology application, vol. 2. IEEE; 2008, p. 70–4.

[16] Rabbi Fazle, Wang Hao, MacCaull Wendy. YAWL2DVE: An automated translator for workflow verification. In: Fourth international conference on secure software integration and reliability improvement. IEEE; 2010, p. 53–9.
[17] Marrella Andrea, Russo Alessandro, Mecella Massimo. Planlets: automatically recovering dynamic processes in YAWL. In: On the move to meaningful internet systems: OTM 2012. Lecture notes in computer science, vol. 7565, Springer; 2012, p. 268–86.
[18] Russello Giovanni, Dong Changyu, Dulay Naranker. Consent-based workflows for healthcare management. In: IEEE workshop on policies for distributed systems and networks. IEEE; 2008, p. 153–61.
[19] Rozinat Anne, Wynn Moe T, van der Aalst Wil MP, ter Hofstede Arthur HM, Fidge Colin J. Workflow simulation for operational decision support. Data Knowl Eng 2009;68(9):834–50.
[20] Nguyên T, Trifan L, Désidéri JA. Resilient workflows for cooperative design. In: Proceedings of the 2011 15th International Conference on Computer Supported Cooperative Work in Design (CSCWD). 2011. p. 69–75.
[21] Adams Michael, Ouyang Chun, ter Hofstede Arthur HM, Yu Yang. Design and performance analysis of load balancing strategies for cloud-based business process management systems. In: Panetto H, et al., editors. 26th internation conference on cooperative infomation systems. Lecture notes in computer science, vol. 11229, Valletta, Malta: Springer Verlag; 2018, p. 390–406.
[22] Adams Michael, Suriadi Suriadi, Kumar Akhil, ter Hofstede Arthur HM. Flexible integration of blockchain with business process automation: A Federated Architecture. In: 32nd international conference on advanced information systems engineering forum (CAiSE'20 Forum). Grenoble, France. 2020.