Wizard, WeMash, WADE: unleash the power of collective intelligence

Luca Trione, Daniela Long, Danilo Gotta, Giovanna Sacchi
Telecom Italia
Via Reiss Romoli 274
10148 Torino - Italy
+39 011 2288035
{luca.trione,daniela.long,danilo.gotta,giovanna.sacchi}@telecomitalia.it

ABSTRACT
This paper presents an approach to the operational process management based on both the availability of tools for the automation of complex activities, and the availability of mashup software instruments. In this way, people directly involved in the operational processes are not only supported in their activities by means of the exiting tools, but they are also allowed, using mashup instruments, to create by themselves new applications, in front of new requirements arising from a changeable context. This paper sketches the problem, presents the adopted solution and describes two applications, currently deployed by Telecom Italia in the Operations Support System domains. Such applications are based on WADE (Workflows and Agents Development Environment), a software platform for the development of distributed applications based on the agent oriented paradigm and exploiting the workflow metaphor to define system logics. WADE is the main evolution of JADE, a popular Open Source framework for the development of multi-agent systems. The first application, called "Wizard", provides a step-by-step guidance both to the on field technicians and to the back-office staff. The second one, named "WeMash", is a mashup platform whose target is to enable 'non developer users' to self-create simple applications and share these applications within the team they are working in.

Categories and Subject Descriptors
I.2.11 {**Artificial Intelligence**]: Distributed Artificial Intelligence - *Multiagent systems*; C.2.4 {**Computer Communication Systems**]: Distributed systems; D.2.11 [**Software Engineering**]: Software architecture

General Terms
Management, Performance, Languages. Human Factors, Standardization.

Keywords
Software Agent, workflow, JADE, Open Source, XPDL, OSS, Telecommunication network, mashup, WEB 2.0 .

Cite as: Wizard, WeMash, WADE: unleash the Power of Collective Intelligence, Luca Trione, Daniela Long, Danilo Gotta, Giovanna Sacchi, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009),* Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. 53 – 60

1. INTRODUCTION
The Telecom Italia is the number one carrier in Italy and one of Europe's fixed-line telecommunications leaders, supplying 21 million telephone lines and over 7.9 million broadband connections in Italy [1], and 2.5 million broadband connections in Holland and Germany.

Also in the mobile arena Telecom Italia is the Italy's leader with around 35.3 million customers, of whom 6.8 million use UMTS technology. Beyond Italy's borders, TIM Brasil is the number two player with a market share of around 25%, corresponding to 35.3 million lines.

To balance the decreasing revenue related to traditional telephony services, Telecom Italia is trying to catch growth opportunities in innovative services and in new addressable adjacent markets and at the same time to improve the efficiency of processes and to reduce the operational costs.

The Group's easy-to-understand, innovative broadband solutions have persuaded many customers to sign up for flat-rated packages, which now account for 76% of all customers. The company supplies over 1.8 million VoIP lines, corresponding to 27.3% of all retail broadband lines. The IPTV service continues to gain ground on the consumer market. Enhanced content and services over the web have attracted an overall portfolio of customers, while web offerings and operations continued to expand. Moreover Telecom Italia started the deployment of the passive optical fiber network (GPON) in combination with VDSL2 modulation enabled the provisioning of more advanced services such as high definition television.

Also in mobile telephony market, spearheading the Group's mobile telephony operations are interactive services and mobile broadband, which are driving VAS revenue growth (VAS now represent 24% of revenue from services).

To provision and configure new services to customers and remove failures and malfunctions Telecom Italia employs about 5,000 back office operators and about 9,000 on feed technicians. These people are the front end of the company towards the customers and in a scenario like the one described above they face:

- the growing complexity and dynamism of the O&M (Operations & Maintenance) activities (e.g provisioning of a new service to customers, troubleshooting and resolving of troubles, …);
- the increasing attention to costs;

- the more and more strong orientation to efficiency, effectiveness and responsiveness to customer needs.

In this scenario it's easy to understand the importance of operation support systems aimed not only at supporting activities of both operator's workforce and customers but also at exploiting and sharing the individual knowledge of technical teams to improve operational processes and speed up the response time to the customer's requests.

These operation support systems enable the reuse of the individual knowledge that so becomes a company asset and a lever for increasing efficiency and effectiveness of operational processes, for reducing costs and for improving customer satisfaction.

In Telecom Italia we are pursuing this goal by implementing two complementary approaches:

- a centralized approach: the knowledge is centrally managed;
- a distributed approach: the knowledge is managed in a distributed way.

Both approaches use WADE (Workflows and Agents Development Environment) [2, 3] as common software platform and both represent the individual knowledge by using workflows; WADE is the main evolution of JADE [5,6,7,8], a popular Open Source framework that facilitates the development of interoperable intelligent multi-agent systems. However, the centralized approach is a more traditional approach that envisages a separation between the owner of the knowledge and the people in charge for the definition of the workflows. Instead, the distributed approach is a more innovative approach in which the owner of the knowledge may also define the workflows. As presented in the following the two approaches are aimed at answering different needs and can also be used in different steps of the knowledge management process.

The paper is structured as follows: in chapters 2 and 3, we present two applications that implement the two different approaches mentioned above an that have a direct influence on the work of thousands of technicians and back office staff. The first one, known as "Wizard", leads operational people in doing their activities;, the second one, named WeMash, allows 'non developer' users to self-create simple applications. In chapter 4 here is a brief description of the WADE platform, used by Wizard and WeMash and in chapter 5 we focus on convergence between centralized and distributed approach. Finally, in chapter 6 and 7 we draw some conclusions and sketch out future activities.

2. Wizard, a way to centralize and use knowledge to lead operational processes

The centralized approach to knowledge management is aimed at automating complex activities in order to:

- broaden the number of people that are able to perform the above mentioned activities;
- empower on field technicians, by increasing their autonomy;
- introduce a standard and optimized way of performing the activities.

The Wizard system, described in [3,4], is a way to carry out the centralized approach; Wizard is a mission critical application deployed by Telecom Italia in the Operation Support System domain. The system provides a set of functionalities that automate complex O&M activities performed by back office staff and on field technicians; the operational knowledge related to these O&M activities is represented by using formal tools (such as workflow). Each functionality, that automate a single O&M activity, both guides the workforce in executing manual steps (like visual controls or physical actions on equipments) and automatically interacts with the OSSs, involved in the O&M activity (e.g. to change a service/network configuration). An example of a complex O&M activity is the ADSL diagnosis and repair activity: the corresponding Wizard functionality leads on field technicians through all steps of the activity.

As previously said, Wizard is based on WADE and therefore each workflow in Wizard is performed by an agent.
A distinguishing characteristic of the WADE workflow engine is the **Delegation mechanism** that allows a set of agents to cooperatively execute a complex process. More in details the agent performing a workflow can decide to delegate the execution of a specific task (always modeled in WADE as a workflow) to another agent on the basis of conditions evaluated at runtime. Such conditions may be related for instance to specific abilities required to carry out a portion of the whole process. In this way Wizard can take advantage of the agent programming paradigm, when it perfoms complex tasks requiring the cooperation of several agents.

To define a Wizard workflow, people use a graphical development environment named WOLF (WOrkflow LiFe cycle management environment); WOLF is not only a tool to graphically create workflows, but a complete environment to manage the whole life cycle of workflows: from the development to the testing up to the deployment. This tool is very powerful, but requires programming skill to be used: so, as the users of Wizard system do not have these skills, they are not able to self-define the workflows that automate their O&M activities. Therefore the definition of Wizard workflows concerns people with two different roles: the operational teams (back office staff and on field technicians), that are the owners of the operational knowledge represented by the workflows and also the users of the Wizard system, and people with programming skill, that define the workflows based on the knowledge provided by the operational teams.

However, due to the compliance of the Wizard system with a centralized approach to knowledge management, another actor is involved in the workflow definition process: this actor is in charge of both the definition of the best practices, based on the collected operational knowledge, and the assessment of these best practices.

The process of collecting and assessing the operational knowledge and sharing it, by using the Wizard system, then entails the following steps:

- collection of the operational knowledge about actual O&M activities;
- identification of the activities to be automated; the selected activities should preferably be:

  - time-consuming;
  - definable in a standard way;
  - cross-functional;
- definition of the best practices for the identified activities;
- assessment of the best practices with the operational people;
- definition of the Wizard workflows corresponding to the reviewed best practices (by using WOLF);
- test of the Wizard workflow (both in a testing environment and in a trial environment);
- deployment of the workflows, that become then available to the target operational teams (back office staff and/or on field technicians).

The centralized approach to knowledge management, above described and carried out by the Wizard system, involves the following advantages:

- standardization and optimization of the O&M activities performed by the operational teams, making them compliant with best practices;
- management of activities with complex logics, through a powerful tool (WOLF) for the definition of workflows representing these activities;
- capability to deal with heterogeneous information models (semantic and syntactic mapping);
- ability to interact also with legacy systems that do not expose their services as Web Services.

However the centralized approach suffers from a few drawbacks:

- it is time-consuming;
- the involvement of people with programming skill is mandatory.

3. WeMash, a way to bring out and manage individual knowledge

In the field of pop music, a mashup is a new song created by mixing vocal and instrumental tracks from different songs. Bringing the concept to informatics world, a mashup is a combination of contents or services from one or more applications, which creates a new application that can include topics, information and services from several different sources. Mashup combines some of the most innovative features of Web 2.0: the vision of the network as a platform where, thanks to shared protocols, applications are able to communicate with each other, the tendency on the part of best known web sites to make contents and services freely available, and the participation of users, who increasingly become independent producer of contents and services. The result of a mashup can be unique and unprecedented, especially for the opportunity to mix different information types, and it can be really innovative if you make available tools that allow an easy mashup design and creation also to users with minimal technical knowledge. Operational processes are evolving and changing so rapidly that it's always more difficult to adapt real time operation support systems: this is an actual challenge for IT departments. Another problem for IT departments is that heterogeneous and ever-changing needs of people involved in operational processes makes really impossible to meet all the requirements and suggestions and very difficult to develop workflows to correctly automate all the operational activities. This implies that for a lot of critical activities and issues every operator/technician or groups of operators/technicians often finds good solutions and implement them by themselves: unfortunately these solutions can be used only by a restricted and small number of users and that means that we are not be able to share and spread individual knowledge.

Mashup and particularly WeMash are a first attempt to solve the problem. WeMash is a Web-based platform whose target is to qualify 'non developer' users to self-create simple applications. Using WeMash a user, involved in provision or maintenance processes, so a user that knows these processes and knows related operation support systems, and that has a particular need, can create his own application/mashup by using available Web services. Having created the mashup, the user can test it and if it runs correctly he can adds information to explain the goal of the mashup and how to use it; then he may ask to a user with an approver profile to test the developed application and to publish it. When the mashup is public all the authorized users can use it. Usually a mashup allows users to automate some repetitive and time consuming work and therefore involves some advantages:

- a lot of users can use the public mashups;
- a lot of users save time to do their work;
- the individual knowledge is now a shared and re-usable knowledge and it's managed in a distributed way;
- operators/technicians build their mashups without any new request to IT department that can allocate more resources on critical projects and on deploying standard Web service as in scenario described in Figure 1;
- users can rapidly build mashups, that cover temporary and urgent needs, and can also create applications to automate regional and specific processes.

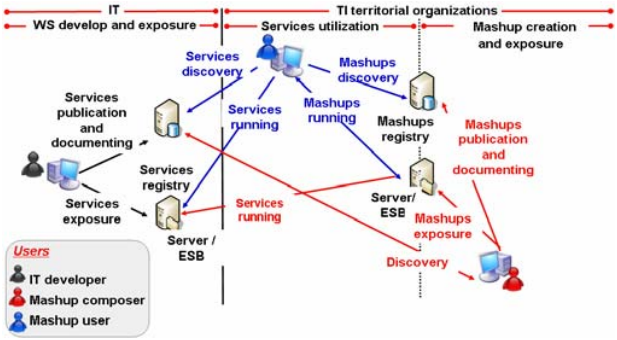

Figure 1 Telecom Italia mashup scenario

WeMash matches some Telecom Italia basic requirements:

1. WeMash is a Web and browser based application and it doesn't require any software or plug-in download. It means that users can create and run their own application or use published mashups directly from their browser.

2. WeMash is absolutely zero code, that is users, supported by software, can create and run their own application or use published mashups using a graphical interface without writing any code. In this way a lot of users without particular programming skill may become mashup creators.

3. WeMash is context-independent and SOA-enabled, that is the application allows users to mix SOAP Web services by simply accessing a Web Services registry. It isn't relevant if web services are related to maintenance or provision or billing or other processes.

4. WeMash is integrated with a Content Management System. In this way it's possible to easily manage an approval workflow for mashups, share them and manage, for example, a lifecycle for mashups. Approved mashups are published on a Web portal, so users, in a Web 2.0 similar approach, can vote and report the best or more useful mashups or can edit and improve existing mashups.

Obviously, WeMash can really improves the capability to rapidly change and control operational processes only if it is coupled with a SOA infrastructure. In other words WeMash requires that the company has deployed a SOA mediation layer, that manages Web Services (exposed by a Universal Description Discovery and Integration registry), and provides policy manager capabilities and intermediation functions. In a SOA scenario we are able to build workflows, compose Web Services and monitor their execution and, using Agent technologies, the entire process should be supported by a distributed trust management [15,16,17,18,19]. The platform architecture includes 3 macro levels, each addressed to a different user profile.

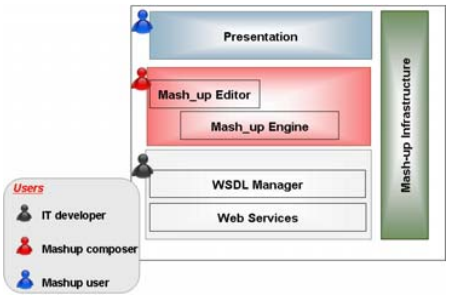

Figure 2 WeMash platform architecture

Mashups user through his browser, accesses a registry of published mahups and user friendly Web interfaces allow him to run mashups and to perform the required interaction operations (input, display, output of data) to correctly execute the application. The mashup composer accesses a Web mashups editor that displays available Web services and supports the mashup composer by establishing relationships between the parameters of services; moreover, the web mashup editor makes available a playground area for achieving the target mashup using easily drag&drop functions. IT developer only provides Web Services.

To provide a concrete example of use of WeMash we can focus a use case related to checks on ADSL circuits. To complete checks, Telecom Italia back office operators usually access different operation support systems using specialized user interfaces and manually exporting and importing data between different interfaces.Instead, composing Web services using Wemash, back office operators can easily create a tool that automatically runs a work flow that manage the different queries to involved operation support systems and the data exchange between the same systems.

In Figure 3 we present an example of the mashup editor related to the scenario descibed above. The mashup composer builds the application simply dragging activities on a canvas area and linking activities to existing Web services. Moreover the mashup composer can manage each input parameter of the activities and he can choose to insert manually the parameter value or to bind the parameter to an output of a previous activity.

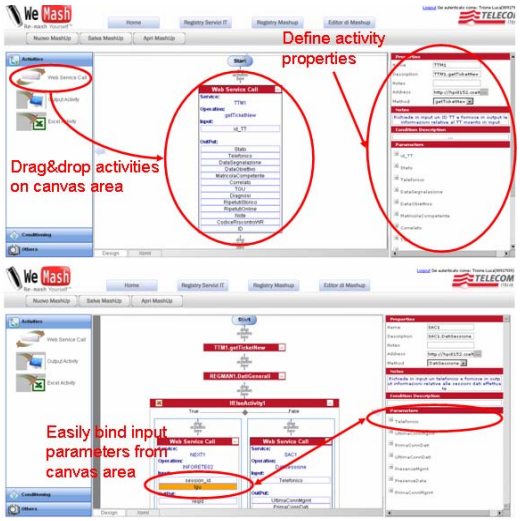

Figure 3 WeMash editor environment

In Figure 4 we present some screen shots related to Wemash execution enviroment and WeMash final activity report, that display to user all the parameters values calculated during mashup execution.

Figure 4 WeMash execution environment

The distributed approach to knowledge involves advantages, as described above, but suffers from a few drawbacks:

- the need to preserve the simplicity of use and to extend the use to a large number of people makes impossible to implement particularly complex processes and manage activities with complex logics;
- it's difficult to deal with misaligned and heterogeneous information models. In particular, sometimes Web Services interoperability could be difficultly guaranteed both at sematic and syntactical level. In order, to cope with this problem as far as the network domain is concerned the SID [28] information model has been adopted in Telecom Italia context.

WeMash system has been developed on top of WADE that provides the support for the execution of tasks defined according to the workflow metaphor (in this way every mashup is described in a graphical form that is understandable by domain experts as well as by programmers) and a number of mechanisms that help managing the complexity of the distribution both in terms of administration and fault tolerance. Moreover the middleware WADE by using a combination of agents and workflows achieves high flexibility in defining and modifying services, high performance and scalability and high control and maintainability on the logics used in the platform.

4. WADE, a common run time platform

WADE (Workflow and Agent Development Environment) is a domain independent platform, built on top of JADE [5], a famous open source middleware [11,12] for the development of distributed applications based on the agent-oriented paradigm. In particular, WADE adds to JADE the support to the workflow execution and a few mechanisms to manage the complexity of the distribution, in terms of administration and fault tolerance.

The systems Wizard and WeMash, described in chapters 2 and 3, are aimed, on the one hand at centralizing and using the knowledge to lead operational processes, on the other hand at bringing out and managing the individual knowledge. However, because of their expressiveness, workflows are used as knowledge representation formalism in both cases. In fact, the main advantage of using workflows, as knowledge representation formalism, is that a workflow can be represented in a graphical form, easily understandable by domain experts as well as by programmers.

WADE exploits the workflow metaphor to define systems and applications logics. Several descriptive formalisms, such as XPDL, BPEL, WS-BPEL [20,21,22,23,24,25,26,27], are available to define workflows, but, although they provide a clear and intuitive representation of the execution flow, they are not always suitable to specify all the details involved in the implementation of an operational process. Such details can be, instead, easily managed by the usually programming languages that are definitely more powerful and flexible to deal with data transformations, computations and other low level auxiliary operations. The approach, followed by WADE, tries to keep both the advantages of a visual and intuitive representation and the power of programming languages in such a way that a workflow, directly implemented by "on field" technicians or back-office staff, can be further developed and detailed, in order to become a new application, supporting operational process automation.

Using WADE as a common runtime platform for Wizard and WeMash, it has therefore been possible to bring out the individual knowledge, to share this knowledge within the "on field" and "back-office" teams and finally to use it to lead operational processes.

Moreover, WADE offers several features, coping with the requirements of flexibility and scalability that are ever more pressing in the OSS domain.

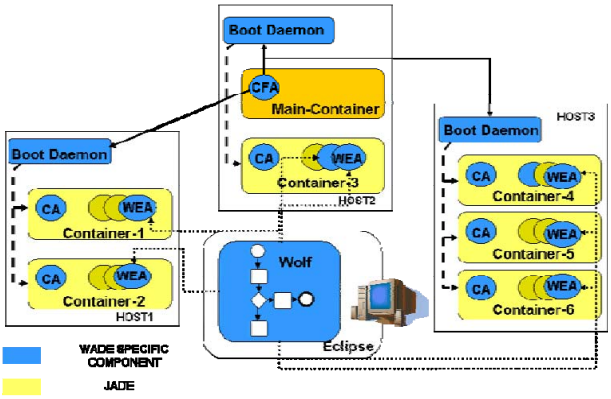

Figure 5 A WADE Platform

As previously mentioned, WADE adds to JADE the support to the workflow execution and a few mechanisms to manage the complexity of the distribution, in terms of administration and fault tolerance. Figure 5, , shows the topology of a WADE-based application where the WADE specific components are highlighted in blue and in particular:

- The **BootDaemon process**: a process running in each host of the platform that it is in charge of the Containers activation within its local host.
- The **Configuration Agent** (CFA): an Agent running in the Main Container and is responsible for interacting with the boot daemons and controlling the application life cycle.
- The **Controller Agents** (CA): there is a controller agent for each container in the platform and they are responsible for supervising activities in the local container and for all the fault tolerance mechanisms provided by WADE.
- The **Workflow Engine Agents** (WEA): they are the most distinguished components of a WADE-based application. In fact, instead of providing a single powerful workflow engine, WADE gives to each JADE agent the possibility of executing workflows and the Workflow Engine Agents are the agents enabled to the workflows executions, by means of an embedded micro workflow engine.

As previously said, WADE tries to make available both the expressiveness of a visual representation of workflows and the power of usual programming languages. In order to do that, it provides a graphical representation of a workflow on top of a normal Java class, that it means that a workflow is implemented directly as a Java class with a well defined structure.

Even if no intermediate formalism is used, WADE adopts the meta-model defined by the XPDL, standard specified by the Workflow Management Consortium [21]. The XPDL meta-model has been chosen, because the XPDL language has been conceived as interchange formalism between different systems. WADE supports the import of XPDL files and the adoption of this meta-model facilitates these operations. In particular, WADE supports all the elements required by the version 1.0 of the XPDL specification and some elements, related to the events, from the version 2.0.

In the XPDL meta-model a process is represented as a workflow, consisting of one or more activities that can be thought as tasks to be executed. In a workflow, the execution entry point is defined, specifying the first activity to be performed; this activity is called **Start Activity**. On the other hand, a workflow must have one or more termination points, named Final Activities.

The execution flow is defined by means of transitions. A transition is an oriented connection between two activities and may have a condition associated. Excluding the final ones, each activity may have one or more outgoing transitions. When the execution of an activity is terminated, the conditions associated to its outgoing transitions are evaluated. As soon as a condition is verified the corresponding transition is activated and the execution flow proceeds towards the destination activity.

Normally a process execution uses some internal data, for instance, to pass intermediate results between activities and/or for evaluation of conditional expressions. In the XPDL meta-model internal data are modeled by **Data Fields**.

A process can have one or more inputs to be provided and one or more outputs expected at the end of its execution. Inputs and outputs of a process can be formalized in the XPDL meta-model by means of the workflow **Formal Parameters**.

Finally, as shown in Figure 5, another important element in the WADE approach to workflows representation is WOLF (WOrkflow LiFe cycle management environment). WOLF is a graphical development environment for WADE-based applications. As its name suggests, WOLF is not only a tool to graphically create workflows for the WADE platform, but a complete environment to manage the whole life cycle of workflows from the development to the deployment. WOLF is an Eclipse [20] plug-in and as a consequence allows exploiting the full power of the Eclipse IDE plus additional WADE-specific features. Moreover, as far as the workflows graphical represention is concerned, WOLF adopts the BPMN notation [25] for both the elements coming from the XPDL 1.0 and the elements deriving from the XPDL 2.0.

5. The evolutionary path: convergence between centralized and distributed approach

A distributed approach to knowledge management, such as that made available by tools like WeMash, described in chapter 3, enables technicians and back office staff, without programming skills, to self develop simple applications. In this way these operational people are able to self-solve in real time contingent issues, covering a good percentage of their needs without requiring any effort from IT department.

Obviously WeMash is not able to answer to all needs arising from operational people; the tools matches a 'zero code and easy to use' paradigm, that allows to quickly develop and share simple applications, and not a 'you can do all with mashup platform' paradigm.

When users need to manage complex processes and activities with embedded complex logics and heterogeneous data models a 'zero code and easy to use' approach is not so effective. In this case a developer, supported by powerful developing tools, should create and manage workflows that automate activities and lead technicians in a more easy and effective way. Hence Telecom Italia has chosen to apply both centralized and distributed approach, to manage knowledge and support in the best way operational processes. The common run time environment for both WeMash applications and Wizard workflows is the WADE platform that provides the workflow engine to run both simple mashups and complex workflows: so we have two development environments, to catch the needs of users with different profiles, and a single run time environment .

The aforementioned development environments are distinct, but not completely separated: there is a relationship between them, better detailed in the following.

As pointed out in session 2 WADE comes with a development environment called WOLF that is an Eclipse plug-in and, as a consequence, allows WADE developers to exploit the full power of the Eclipse IDE plus additional WADE-specific features and facilitates the creation of WADE-based application.

In our scenario (Figure 6) we make available a software platform (Session 5.1), that provides technicians and operators with the capabilities to easily create mashups and IT developer with the capabilities to import mashups, as workflows, in WADE development environment. Using WOLF developers can easily manage, improve and complete mashups to create and test complex business solutions ready to be distributed to users.

Figure 6 Wemash&WIZARD/WOLF integrated scenario

In this way the following efficiency and effectiveness objectives can be achieved:

- users rapidly self creates and shares applications to automate activities and in this way manages in the best way new activities and new operational processes;
- IT department acquires in a very easy and clear way software requirements from operational teams because created mashups are, in fact, new and clear cut requirements for operation support systems;

- IT department can speed up the time it takes to develop and deploy workflows and new functionalities by improving existing mashups.

5.1 WADE – the evolutionary path to support Wizard/WeMash convergence

As described in the previous chapters, Wizard and WeMash require support for:

- Web Services invocation;
- Interactive workflows.

As it is shown in Figure 7, the WADE platform has therefore been improved, in order to become a suitable common runtime for both WeMash and Wizard.

In particular, as far as the support for Web Services is concerned, such kind of support has directly been added to the WADE platform, improving the micro workflow engine of Wade Engine Agents and making it able to execute a new type of activity, called WebService Activity that is in charge of invoking a Web Service. Also the WOLF editor has been empowered to support the editing of such kind of activities.

Moreover, a new layer has been added on top of WADE that provides all features necessary to manage the user interaction during the workflows execution.

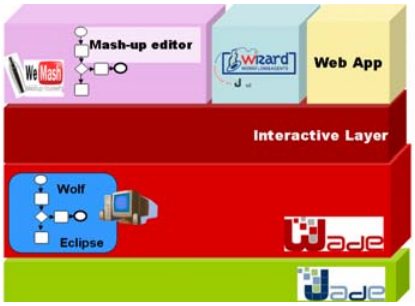

Figure 7 – WADE as a common runtime

6. CONCLUSION

Operational processes are important to the successful operation of an enterprise. BPMS and job management systems provide good support for managing processes, but more advanced approaches are necessary in order to meet the challenges of business agility. We think that mashup joined with a powerful workflow engine and agent technology can be used to provide a flexible, rapid and agile way to model, design and execute controlled operational processes.

This paper presented our approach to operational process management which exploits WADE platform, and particularly WADE Workflow Engine Agents (WEA), as a common platform with the goal to allow skilled users to simply create and share applications. Agent technology enables our platform to ensure high flexibility in defining and modifying operational processes, high control and maintainability of the logics used in and high performance and scalability.

In the paper we have described our experiences in applying the aforementioned approach to the Telecom Italia operational processes. We have briefly depicted the Wizard system, already deployed in field, whose workflows are supportimg field technicians and back-office staff activities. We have also described our mashup platform prototype, named WeMash: the prototype is currently trialed by groups of 'non developer' users and, in parallel, we are working to engineer it in order to deploy it in 2009.

From March 2007 more than 100,000 operational activities have been successfully executed by Wizard with an average time reduction of more than 50% and peaks, in terms of time reduction, of about 80%. As the coverage of Wizard system, with regard to provided functions, is expected to grow, we preview that in 2010 about 1.4 Million operational activities will be executed by Wizard. Furthermore, with WeMash deployment we will spread the capabilities to create useful applications by using the agent platform also to 'non developer' users, hence to the majority of operational staff.

As a reduction of just one minute per day (with reference to the time Telecom Italia operational people spend doing their activities) generates savings of about 1 Million euros, it's easy to understand that much further progresses are expected along the timeline of the overall project.

7. OUTLOOK

Future activities are related to the possibility to expose Wizard workflows as Web Services re-usable with WeMash capabilities.

Another future activity is focused on empowering the WADE support for Web Services adding to them a semantic meaning [10]. In particular, we are investigating how to add semantic meaning to Web Service both for automatic composition and interoperability purposes.

In a mashup context, in fact, a very important issue is to guarantee that a workflow continues working, even if some of Web Services that it invokes are modified or substituted.

Nowadays, in the academic community it is coming out the idea that agents, together with the technologies belonging to the Semantic Web field, can play an important role in semi-automatic/automatic Web Services composition [14, 15, 17, 18 e 19]. In WADE and therefore in WeMash, we are trying to exploit the reasoning capabilities of agents in order to allow the definition of workflows invoking Web Services, just using their semantic meaning. In this way, at run time, the agents will be in charge of discover and match such semantic meaning with a concrete Web Service implementation, overcoming the problems resulting from a strong binding between a workflow and its Web Services implementaions.

8. ACKNOWLEDGMENTS

We gratefully acknowledge all of our international friends who have contributed to JADE over the years. Furthermore we are grateful to our Telecom Italia colleagues (in primis the OSS Innovation department and the Trento Software factory) for the contributions to the systems implementation, deployment, testing and maintenance of WADE e Wizard

9. REFERENCES

[1] Telecom Italia Quarterly Report at September 30, 2008 http://www.telecomitalia.it/TIPortale/docs/investor/TI_Report_Q3_2008_Eng.pdf

[2] WADE – Workflows and Agent Development Environment http://jade.tilab.com/wade

[3] Caire, G., Gotta, D., and Banzi, M. 2008. WADE: a software platform to develop mission critical applications exploiting agents and workflows. In Proceedings of the 7th international Joint Conference on Autonomous Agents and Multiagent Systems: industrial Track (Estoril, Portugal, May 12 - 16, 2008). International Conference on Autonomous Agents. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 29-36. DOI= http://doi.acm.org/10.1145/1402795.1402802

[4] G. Covino, D.Gotta, A.Nasuto, 2007. La gestione delle nuove reti con un diverso paradigma, pp 11-13. http://www.telecomitalia.it/TIPortale/docs/innovazione/012007/Pag27_42_NNEM.pdf

[5] JADE - Java Agent Development framework. http://jade.tilab.com.

[6] Bellifemine F., Caire G., D. Greenwood. Feb. 2007, Developing multi-agent systems with JADE. Wiley Series in Agent Technology. ISBN 978-0-470-05747-6

[7] Bellifemine, F. and Rimassa, G. 2001. Developing multi-agent systems with a FIPA-compliant agent framework. *Softw. Pract. Exper.* 31, 2 (Feb. 2001), 103-128. DOI= http://dx.doi.org/10.1002/1097-024X(200102)31:2<103::AID-SPE358>3.0.CO;2-O

[8] Ughetti, M., Trucco, T., and Gotta, D. 2008. Development of Agent-Based, Peer-to-Peer Mobile Applications on ANDROID with JADE. In Proceedings of the 2008 the Second international Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies - Volume 00 (September 29 - October 04, 2008). UBICOMM. IEEE Computer Society, Washington, DC, 287-294. DOI= http://dx.doi.org/10.1109/UBICOMM.2008.72

[9] G. Caire, M. Porta, E. Quarantotto, G. Sacchi "Wolf - An Eclipse Plugin for WADE", Procs. WETICE workshop ACEC 2008 in press.

[10] Ankolekar, A., Krötzsch, M., Tran, T., and Vrandečić, D. 2008. The two cultures: Mashing up Web 2.0 and the Semantic Web. Web Semant. 6, 1 (Feb. 2008), 70-75. DOI= http://dx.doi.org/10.1016/j.websem.2007.11.005

[11] Lee H., Mihailescu P., Shepherdson J., . 2007, Realising Team-Working in the Field: An Agent-based Approach, IEEE Pervasive Computing, pp. 85-92 .

[12] AgentLink III. Agent Technology Roadmap. Available from http://www.agentlink.org/roadmap/index.html

[13] Belecheanu R., Munroe S., Luck M., Payne T., Miller T., Pechoycek M., McBurney P, 2006 . Commercial applications of agents lessons, experiences and challenges, Industrial Track Fifth international joint Conference on Autonomous Agents and Multi-Agents Systems. ACM Press, pp 1549-1555

[14] P. Mordacci, A. Poggi, C. G. Tiso, P. Turci, Using Agent Technology as a Support for an Enterprise Service Bus. Proc. WOA 2008, pp 5-10 http://www.pa.icar.cnr.it/woa08/materiali/Proceedings.pdf

[15] Poggi A., Tomaiuolo M., Turci P., 2007, An Agent-Based Service Oriented Architecture http://woa07.disi.unige.it/papers/PoggiSOA.pdf

[16] Foster I., Jennings N. R., Kesselman C., 2004, Brain Meets Brawn: Why Grid and Agents Need Each Other, Proc. Autonomous Agents and Multi Agent Systems (AAMAS 2004), pp 8-15

[17] Greenwood, D., Callisti, M. Engineering Web Service-Agent Integration. In IEEE Conference of Systems, Man and Cybernetics, 2004. Available from http://www.whitestein.com/resources/papers/ieeesmc04.pdf

[18] Savarimuthu, B.T.R.; Purvis, Mary.; Purvis, Mart.; Cranefield, S., 2005.,Integrating Web services with agent based workflow management system (WfMS), Proceedings. The 2005 IEEE/WIC/ACM International Conference on Web Intelligence, Page(s): 471 - 474

[19] Negri A., Poggi A., Tomaiuolo M., Turci P,. 2006, Dynamic Grid Tasks Composition and Distribution through Agents,. Concurrency and Computation: Practice and Experience(2006), 18(8): 875-885

[20] Eclipse. www.eclipse.org.

[21] WFMC WorkFlow Managment Coalition, http://www.wfmc.org/

[22] XPDL XML Process Definition Language, http://www.wfmc.org/standards/xpdl.htm

[23] van der Aalst W.M.P.,. 2003,Patterns and XPDL: A Critical Evaluation of the XML Process Definition Language. QUT Technical report, FIT-TR-2003-06, Queensland University of Technology, Brisbane,

[24] Shapiro, R. 2002. A comparison of XPDL, BPML and BPEL4WS (Rough Draft), Cape Vision

[25] BPMN Business Process Modeling Notation http://www.bpmn.org/

[26] P. Wohed, W.M.P. van der Aalst, M. Dumas, and A.H.M. ter Hofstede., 2003 Analysis of Web Services Composition Languages: The Case of BPEL4WS., 22nd International Conference on Conceptual Modeling (ER 2003), volume 2813 of Lecture Notes in Computer Science, pages 200-215. Springer-Verlag, Berlin, 2003.

[27] WS BPEL Web Services Business Process Execution Language Version 2.0, OASIS Standard, 2007, Available from http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf

[28] Tele Management Forum. www.tmforum.org