Automating Marketing Campaign Management Through an Agent-based Workflow Management System

PAVLOS DELIAS[1], KLIMIS NTALIANIS[2], ANASTASIOS DOULAMIS[1], NIKOLAOS MATSATSINIS[1]

[1]Department of Production Engineering and Management
Technical Univeristy of Crete
73100, Chania,
GREECE
pavlos@ergasya.tuc.gr
[2]Electrical and Computer Engineering Department
National Technical University of Athens
9, Iroon Polytechniou str., Zografou 15773, Athens
GREECE
kntal@image.ntua.gr

Abstract: - Marketing campaigns are an important marketing process that support the communication of brands and products to the wide public. Although a campaign may be significantly different from another, the main steps have been outlined by marketing experts, making a formal representation of the campaign workflow possible. In this paper, we advocate that using a process definition language to describe the marketing campaigns, we can support the management of multiple concurrent campaigns through automation. A multi-agent architecture is proposed to frame a workflow management system so that extensive integration of heterogeneous resources is achieved. These resources may refer to humans, software tools, machines or organizational assets, all required for the actual execution of a campaign. The coordination of the resources is based on the agent brokering paradigm and on scheduling algorithms. A prototype architecture, extending the open source platform *WADE*, is summarized and presented in a dedicated section, while an outline of the critical functions is also attached.

Key-Words: - Multi-Agent Systems, Workflow Management, Marketing Campaign

1 Introduction

Marketing campaigns are a major marketing communication mode that includes a broad mixture of tools and activities such as budgeting, forecasting, managing digital assets, and dealing with complex scheduling requirements. Because of the proliferation of products and brands, even larger number of market segments, fierceness of competition, and overall acceleration of change, marketing campaigns have become complex and their planning and administrative decisions must be made under increasing time pressure. Indeed, timing and sequencing activities within a campaign is one of the critical decision variables [11]. A campaign plan calls for more than assigning tasks and checking deadlines. An effective marketing campaign management should also optimize resources allocation, aware for tasks' inter-dependencies and timely respond to the client needs.

For an effective campaign management to happen, we propose in this paper a two-dimensional approach. Firstly, integrate the campaign management process into a workflow management system (WfMS), so that the process is automated as much as possible. Then, leverage methods from production management, to optimize features that are common in both fields, such as resources allocation and activities scheduling.

The rough main activities of a marketing campaign have been analytically described in popular handbooks of marketing [8]. However, it is clear that a campaign could focus on some special steps or it could omit some others, it could execute the steps sequentially or parallelize the process, according to the campaign's special requirements. Moreover, each step may contain different activities in a variety of flows. Human resources may be involved along with machines or organizational assets. All these resources may be related; require particular sequencing, or even impose special constraints (e.g., release times, deadlines; an activity can be executed only by a special resource etc.). In addition,

campaign management (as most marketing problems [9]) requires a bringing together of people ideas, data and judgments from diverse and heterogeneous sources. Because of the above particularities, every campaign may significantly differ from another. This diversity makes the management of multiple concurrent campaigns a difficult process. The overall process can be facilitated by formally representing the campaign processes. Assuming that a formal representation of each campaign is available, an ordinary way to address the problem is through project management techniques [15]. Gantt charts are a popular method to generate a project plan and monitor its execution. Indeed, some vendors (SAP [Table 1], Microsoft: **http://o-ce.microsoft.com/en-us/templates/TC012330891033.aspx? CategoryID= CT102115851033**) provide marketing campaign blueprints so that charting a campaign project and monitoring its workflow is facilitated. Still, even with the use of specialized project management software, campaign management remains a manual operation. A critical reason for this situation is that marketing campaign management requires extensive integration of heterogeneous and possibly distributed resources and a continuous monitoring of the campaign execution.

Table 1. SAP Business Workflow in Campaign Automation. *Source*: http://help.sap.com/saphelp_crm70/helpdata/EN/45/ ffcbced6f771fae10000000a1553f6/ content.htm

Workflow templates for Campaign Automation
WS14000061 Transfer Target Group to Channel
WS14000062 Create Target Group
WS14000062 Create Target Group and Channel Transfer
WS14000064 Send E-Mail to Employee Responsible
WS14000065 Authorization by Employee Responsible
WS14000066 Adding a Business Partner to a Target Group
WS14000067 Deleting a Business Partner from a Target Group
WS14000068 Start Target Group Optimization
WS14000069 Transfer Respondent to Channel
WS14000070 Start Subsequent Step Without Executing
WS15100040 Start Media Campaign

In [20], a multi agent system is used to automate the project management process. Activities are scheduled and resource conflicts are resolved by message exchanging and negotiation among agents. Although this way scheduling and resource allocation issues are effectively addressed, the monitoring of the project needs a more workflow-oriented approach. Anticipating this workflow requirement, Blake in [2] provides some interaction protocols for coordinating agent teams in the business process orchestration domain. More specifically, organizations of agents are collaborating to compose heterogeneous services into executable workflows. Interaction protocols of agents that collaborate to support process specific workflows are also used in [6], [12]. The proposed agent-based WfMS exploits interaction protocols together with a sound process modeling method (Petri-Nets), to enhance the awareness of the system and to better monitor the actions of the workflow participants.

In this study, we follow the agents paradigm to propose a workflow management system that could facilitate the automation of the campaign management. The analytical procedure is described in the next section, while in section 3 the basics of a prototype architecture are discussed. The main functions of the proposed system are presented in section 4 and some further discussion concludes the paper.

2 Problem Modeling

In our work we consider every campaign to be a distinct process. Each process instance represents one individual enactment of the campaign which is capable of independent control and audit as it progresses towards completion. Upon execution, each instance exhibits the campaign's internal state, which represents its progress and its status with respect to its constituent atomic activities. To align our concept with broadly accepted standards, we ask for this process to be formally described by the *XML Process Definition Language* (*XPDL*) [19]. The *XPDL* is a popular workflow process definition language, adopted by numerous vendor organizations, and it is fully operable with the *Business Process Modeling Notation* (*BPMN*) of the *Business Process Management Initiative* (*BPMI*) [10]. To facilitate the end users of the system, a campaign process may be defined by an existing process template. Of course, the end users should be able to modify existing templates or creating new campaign processes from scratch.

This way, a campaign process definition is ultimately an XML document. This is an important advantage as the XML documents can be easily parsed so that a computer can automatically comprehend the process structure. An illustrative example of an *XPDL* document is presented in Figure 1. Following the *XPDL* standard, and with respect to our marketing campaign management context, the main elements of the campaign process definition are:

```
- <Participants>
  - <Participant Id="Participant1" Name="MarCom">
      <ParticipantType Type="HUMAN" />
    </Participant>
  - <Participant Id="Participant2" Name="Printer">
      <ParticipantType Type="RESOURCE" />
    </Participant>
  </Participants>
- <Activities>
  - <Activity Id="Identify_Target">
      <Performer>Participant1</Performer>
    - <TransitionRestrictions>
      - <TransitionRestriction>
        - <Split Type="AND">
          - <TransitionRefs>
              <TransitionRef Id="Transition_2" />
              <TransitionRef Id="Transition_1" />
              <TransitionRef Id="Transition_3" />
            </TransitionRefs>
          </Split>
        </TransitionRestriction>
      </TransitionRestrictions>
    - <ExtendedAttributes>
        <ExtendedAttribute Name="duration" Value="5" />
      </ExtendedAttributes>
    </Activity>
    ...
  </Activities>
- <Transitions>
    <Transition From="Identify_Target" Id="Transition_1" To="Communicate_Director" />
    <Transition From="Identify_Target" Id="Transition_2" To="Communicate_Manager" />
    <Transition From="Identify_Target" Id="Transition_3" To="Publish_TgtDoc" />
    <Transition From="Communicate_Manager" Id="Transition_4" To="Establish_Budget" />
```

Figure 1: Code snippets from an example XPDL document wherein the <Activity>, <ExtendedAttributes>, <Transition>, <TransitionRestriction>, and the <Performer> tags appear.

➢ The atomic activities, which can be seen as the individual steps of the process. The activities are responsible for the actual execution of the campaign and they can be either the smallest units of work which need to be scheduled, or composite activities which contain other sub-activities (in the latter case the activities are actually called "*Subflows*"). Activities may be manual (and thus need humans for their execution), or automated (and thus require software tools to support their execution). In the *XPDL* documents each activity is declared within an <Activity> tag, while its duration is declared by exploiting the <ExtendedAttributes> feature of the *XPDL* and inserting the numerical value at the corresponded attribute.

➢ The transitions among the activities. The transitions define the flow of the process. Virtually any flow is possible as a campaign process can be arbitrarily complex. However, the basic flow structures are the sequence, the parallel execution and the conditional branching. We used the <Transition> tag to record each existing transition in the processes and to address the sequence flows. In addition, we used the <TransitionRestriction> tag to refine the transitions' set and to tackle complex flows.

➢ The performers of the activities, namely the necessary resources (humans or tools) needed to execute the campaign. By definition, in the *XPDL* document, each activity declares its performer using the <Performer> tag. According to the *XPDL* specification, the performers' types are: *RESOURCE_SET*, *RESOURCE*, *ORGANIZATIONAL_UNIT*, *ROLE*, *HUMAN*, and *SYSTEM*. In our study, a performer could be anything among the above types.

We consider a time window *T* when *N* marketing campaigns demand for execution. This time window can be considered as a time interval after which a new scheduling is activated. The system can decompose these processes into atomic activities through the available XML process definitions, so it is equivalent to say that a set of activities demands for execution.

After the XML parsing of the definitions, we are able to form two matrices: The first one contains all the atomic activities that request for execution and their durations. The second one contains the precedence constraints that the atomic activities impose one to another. Although the precedence constraints exist between activities of the same

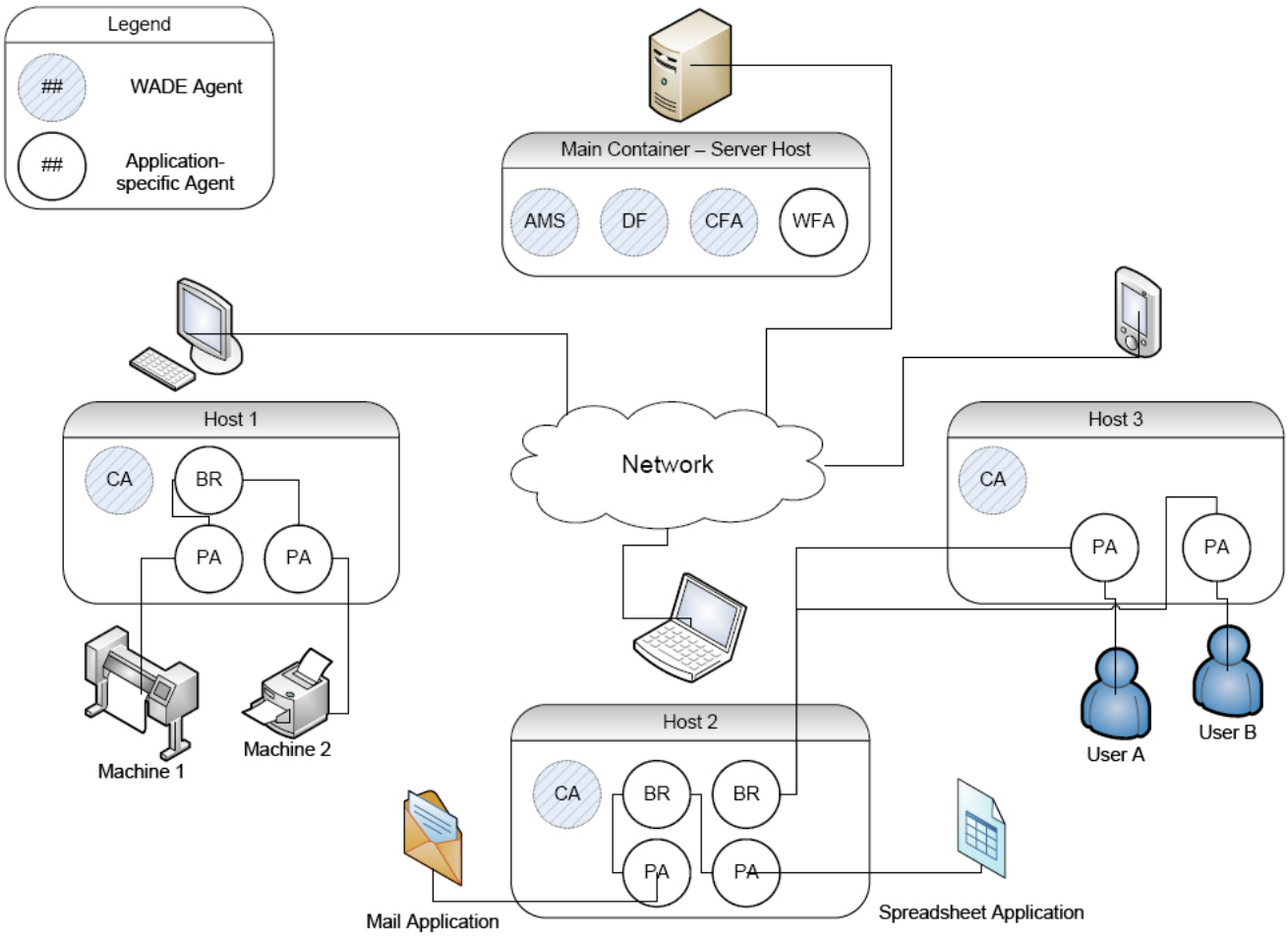

Figure 2: An agent-based architecture

campaign, both matrices contain the global information and they do not separate the activities respective to the campaign they belong.

Our methodology proceeds following a performer-group perspective. More specifically, every performer is classified according to its service property (e.g., a printer; a Director etc.).We consider atomic activities to request not for a specific performer but for a specific group of performers (e.g., a color printer and not the color printer A or B). Then the scheduling is made for each performer group.

Our goal is to allocate the pending atomic activities to the available resources so that a) temporal (precedence) constraints are preserved and b) the overall campaigns completion time is minimized. We claim that a fair scheduling algorithm tackles both the above issues, as it distributes the activities to the available resources. The implementation of a multi-agent system that supports the procedure is described in the following sections.

3 An Agent-based Architecture for Workflow Management

Considering the arguments discussed in the introductory section, we advocate that an agent-based architecture could properly manage the heterogeneous and rather autonomous resources that are involved in a marketing campaign. The general idea is that resources are represented by proxy agents, who are registered with a broker agent. The number of broker agents equals the performer-groups described in the previous section, one broker per group. Each broker is responsible for the coordination of its own group's agents (resources). This concept is quite popular in the WfMS literature [1], [3], [13], [17], [21] as it provides a flexible yet robust infrastructure.

The proposed WfMS is based on *WADE* [4] which provides the rough architectural concept and the main functionalities. The *WADE* context allows the agent-based system to be fully distributed on different containers. The containers themselves may reside on different hosts. The architecture of the

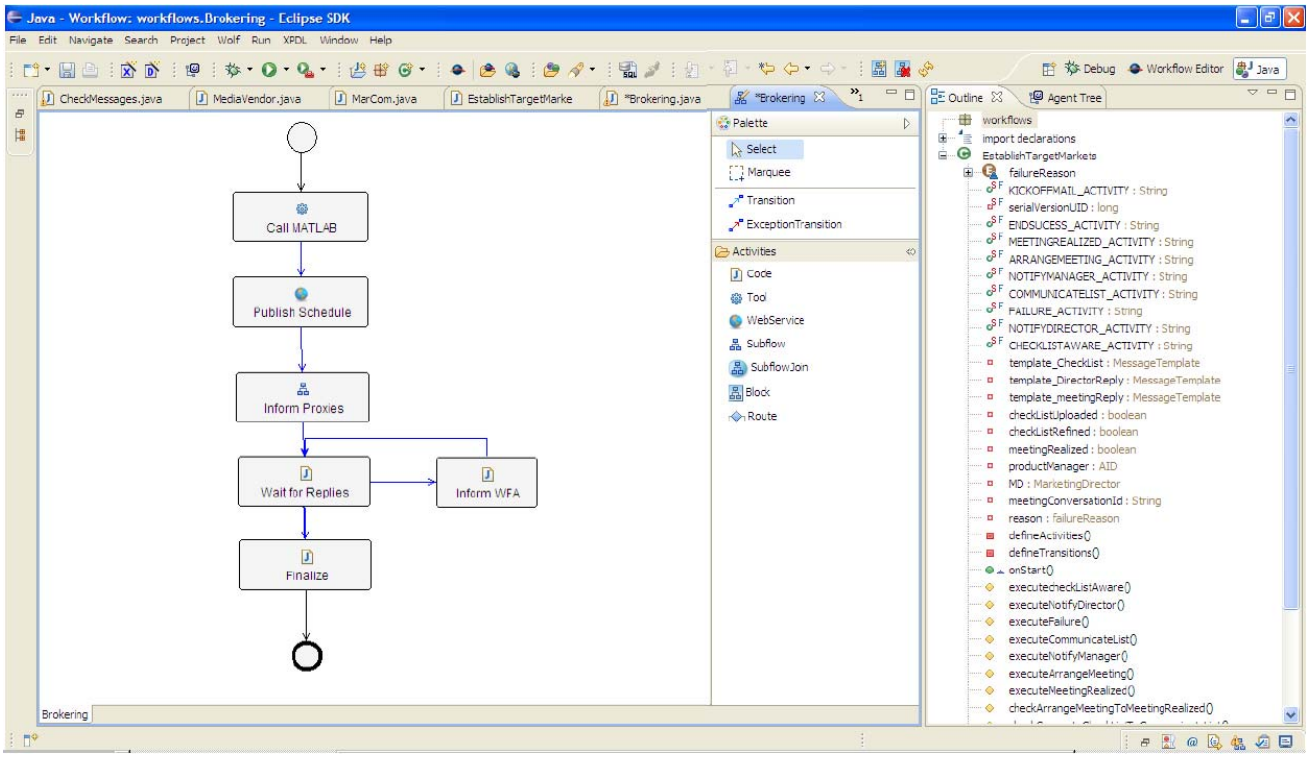

Figure 3: The standard workflow behavior of a broker agent

WfMS we suggest, comprises two kind of agents: the *WADE* agents (that support the administration of the system), and the application-specific agents (who are agents dedicated to the campaign management). The application agents are further distinguished between broker agents and proxies. A proxy agent may represent a machine, a software tool or even a human. It stands for an organization resource and it is in essence an "activity executor". Our method does not impose any special requirements about where every agent should be, as long as it can smoothly communicate with the administrative agents.

Scheduling and resource management are two of the major functions of the workflow engine, the core component of a WfMS [18]. However, in this study we detach these operations from the engine and we assign them to the brokers. Moreover, since a central workflow engine does not exist in the *WADE* platform, we need an extra agent to:

1. Receive the campaigns that request execution
2. Decompose the campaigns into atomic activities
3. Group the activities according to their performers

An outline of the proposed architecture is depicted in Figure 2. The *WADE* agents, necessary to administer the platform are:

- The *Agent Management System* (AMS), which provides white-page and life-cycle services for all the agents registered in the platform. Only one AMS exists in the running platform.
- The *Directory Facilitator* (DF) is the agent who provides the default yellow page service in the platform. Proxy agents register with the DF to publish the service they provide, while broker agents subscribe with the DF to get informed about which proxies are available.
- The *Configuration Agent* (CFA), which is responsible for interacting with every host and controlling the marketing campaign management life cycle. CFA together with DF and AMS are always running in the Main Container.
- The *Controller Agents* (CA), one per container. CAs are responsible for the supervising activities in the local container and for all the fault tolerance mechanisms provided by *WADE*.

As far as the application specific agents are concerned, in the Main Container lies the Workflow Central Agent (WFA) which receives a batch of campaigns requesting for execution; parses the *XPDL* documents; decomposes the campaigns into atomic activities, and clusters the activities according to their performers. The proxy agents (PA), which represent the resources of the system (no matter the resource type, e.g., tools or humans) are distributed in the platform. Broker agents (BA) are also distributed in the platform, and they may or may not be in the same containers with the PAs they

supervise. Broker agents know what agents they supervise because of the FIPA Subscribe Interaction Protocol that they implement. More specifically, upon initialization, a proxy agent registers with the DF, declaring the service it provides. Every broker, upon its own initialization, subscribes with the DF, requesting to get informed every time a PA of a speci ed service registers or deregisters. The DF communicates any changes may occur, thus keeping up-to-date all brokers. Agents' communication is message-based and FIPA-compliant, and it is provided by *WADE* as a standard facility.

4 Functional Scenaria of the Proposed System

4.1 The Broker Agent

The broker agent embeds a small and lightweight workflow engine and it is capable of executing workflows represented according to a *WADE* specific formalism. These workflows are significantly different from the workflows that describe the campaigns processes, since the *WADE* workflow representation formalism is based on the JAVA language. Actually, the workflows that are executed by the brokers are ultimately JAVA classes. Although the *WADE* platform allows for dynamic loading of workflow classes, in our system, the main behavior of brokers is based on a fixed workflow process, illustrated in Figure 3.

The execution of this workflow is triggered when a new message that contains the set of activities that request for execution arrives from the WFA. This workflow process dictates the broker to perform a set of tasks, according to a specific flow. In particular, the broker calls a MATLAB engine to solve the scheduling problem, according to a pre-specified algorithm [see 4.2]. MATLAB returns a matrix with the solution, and a plot with a graphical representation of it. The plot is captured as an image by the broker and it is published through a Web Service to an internal web page. The next step is to inform the proxies. This is a composite activity that contains the following steps:

- Iterate over the activities that are described in the scheduling matrix (MATLAB result). For each activity
  1. Create a Message to be sent to the agent that the activity will be assigned. This message's perfomative is set to *REQUEST* [7].
  2. Set the content of the message as the identification code of the activity
  3. If the activity has any precedent activities then add to the proxy's precedents map the corresponding actvities' records. Every precedents map record contains a key and a value. The value is the identification code of the activity and the key is the identification name of the agent that is responsible for its execution (*AID* in *WADE* terminology)
- Wait for the confirmation messages from every proxy agent. The confirmation messages are messages of *AGREE* perfomative, and match the conversation Id that the broker specified in the previous step. This way, they are easily distinguished by brokers.
- When all confirmations have been collected, then the broker performs some finalization activities and puts itself into the *WAITING* mode (its internal thread is sleeping).

4.2 The scheduling algorithm

In our work we use the *TORSCHE* toolbox [16] to resolve the scheduling problem. The *TORSCHE* toolbox provides a set of scheduling algorithms, however in our application we used the List Scheduling algorithm [14] to address the problem we described in section 2. The input data that the List scheduling algorithm requires are obtainable from the *XPDL* documents that describe each campaign and these are: The set of tasks that are pending, their durations and a square matrix with their precedence requirements.

The algorithm we used is a simple heuristic that optimizes the overall completion time. Although that this algorithm has a low complexity ($O(n)$), and it works effectively in an agent-based context where message exchange is intense and obstructs the strict timelines of a schedule, it is clear that any other scheduling algorithm could support the system we propose as well. Indeed, in the case that there is a heavy load of pending activities, more efficient techniques should be used [5]. The final output of the algorithm is a sequence of the tasks that should be executed per proxy agent. These results are returned to the broker agent in a matrix format and to a graphical format (Figure 4) as well.

4.3 Interactions

A major part of the system's functionality is based on the message exchange among the participating agents. To assure the soundness of these message-based interactions among agents, we took advantage of the Interaction Protocols specified by FIPA [7]. In particular, brokers communicate with the DF implementing the Subscribe Interaction Protocol,

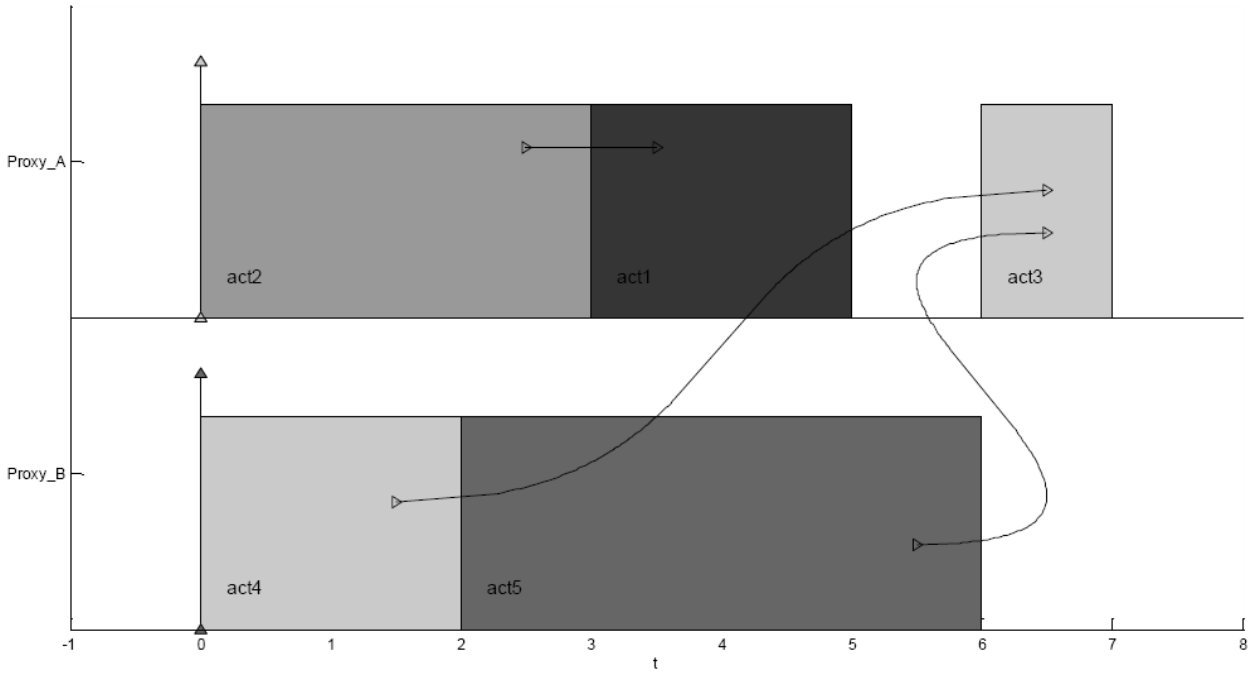

Figure 4: A schedule plot, as captured by a broker agent, which represents two proxies.

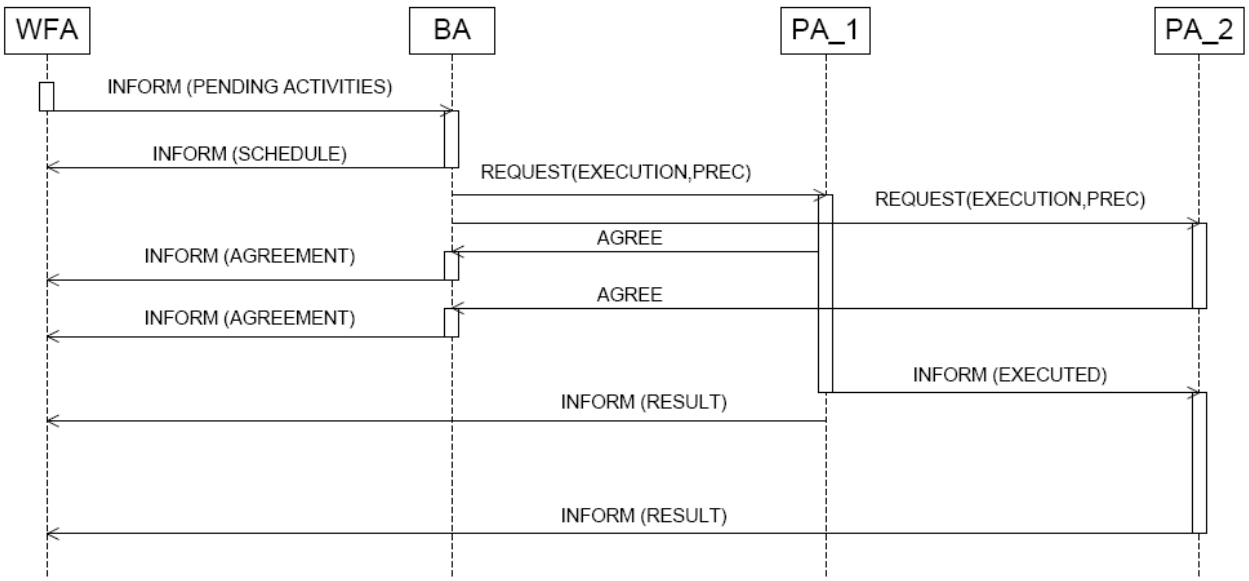

Figure 5: Critical interactions of the application-specific agents.

while the activities assignments, between the brokers and the proxies they represent, follows the Request Interaction Protocol.

The core procedure interactions that take place in an example case where the platform contains a broker (BA) and two proxies registered with him (PA_1, PA_2) are illustrated in Figure 5. Each arrow indicates a message that is exchanged while the text above the arrow indicates the message's perfomative. In the parentheses we put a meaningful word of the content of the message. Of course, Figure 5 illustrates a selective portion of the messages being exchanged between agents, since the *WADE* administration is also message-based to a great extend. Agents are able to distinguish messages using a message template, defined by the users of the platform. Moreover, *WADE* provides the opportunity to an agent to block its execution until a message that matches a specified template is received. This way, sound interaction protocols are achieved and seamless message-based communication is accomplished.

5 Conclusion

In this paper, we presented a prototype multi-agent workflow management system to automate the marketing campaigns management. We recast campaigns in terms of business processes, by describing them using a formal process definition language. This way, the campaign process structure can be interpreted by a computer and lead to effective scheduling and monitoring. The workflow support that the proposed system provides, advances the project management methodologies, often used to carry out the same work. However, when it comes to the application of our work to real world scenaria, there are two critical points that must be noticed.

First, we assume that a formal representation of the campaign process exists. If this is not the case, or if the representation is not accurate, it is clear that the solution proposed by the system will neither be sound or effective. Second, the autonomy of the human resources is restricted by the autonomy level of their proxy agents. That means, that a human should deliver what exactly he is being asked, and that he can not modify any parameters of the deliverable. This situation allows a stricter monitoring of the campaign progress, but it is not always the desired situation.

Although in this paper we suggest the multi-agent system to be used for campaign management, the system could be easily expanded to manage the workflows of other concurrent processes too (not related to marketing campaigns). Another future direction of this research is to develop an integrated planning and scheduling algorithm so that an exact process definition will not be necessary. Considering that an effective planner could generate a robust process model, we can limit the input requirements just to the list of the activities and to their precedence requirements, eliminating the necessity of a formal representation of the processes.

6 Acknowledgement

This study is supported by the Special Research Fund Account (ELKE) of the Technical University of Crete, under the project EYSYMA: An intelligent cooperative marketing support system.

References:

[1] Thandar Aye and Khin Mar Lar Tun, “A collaborative mobile agent-based workflow System,” In *6th Asia-Pacific Symposium on Information and Telecommunication Technologies* (APSITT 2005), pages 59-65, Yangon, Myanmar, 2005.

[2] M. Brian Blake, “Coordinating multiple agents for workflow-oriented process orchestration,” *Information Systems and E-Business Management*, 1(4):387-404, 2003.

[3] M. Brian Blake and Hassan Gomaa, “Agent-oriented compositional approaches to services-based cross-organizational workflow,” *Decision Support Systems*, 40(1):31-50, 2005.

[4] Giovanni Caire, Danilo Gotta, and Massimo Banzi, “WADE: a software platform to develop mission critical applications exploiting agents and workflows,” In *AAMAS '08: Proceedings of the 7th international joint conference on Autonomous agents and multi-agent systems*, pages 29-36, Richland, SC, 2008. International Foundation for Autonomous Agents and Multiagent Systems.

[5] Pavlos Delias, Anastasios Doulamis, and Nikolaos Matsatsinis, “A joint optimization algorithm for dispatching tasks in agent-based workflow management systems,” In Jose Cordeiro and Joaquim Filipe, editors, *Proceedings of the 10th International Conference on Enterprise Information Systems* (ICEIS 2008), pages 199-206, Barcelona, Spain, 2008.

[6] Lars Ehrler, Martin Fleurke, Maryam Purvis, and Bastin Tony Roy Savarimuthu. “Agent-based workflow management systems (WfMSs),” *Information Systems and E-Business Management*, 4(1):5-23, 2006.

[7] FIPA. Fipa communicative act library specification. Technical report, *FOUNDATION FOR INTELLIGENT PHYSICAL AGENTS*, 2002.

[8] Philip Kotler and Kevin Lane Keller. *Marketing Management*. Pearson Prentice Hall, Upper Saddle River, New Jersey, 2006.

[9] J.D.C. Little, “Models and managers: the concept of a decision calculus,” *Management Science*, 16(8):B4, 66-85, 1970.

[10] OMG. Business process modeling notation, 2009. http://www.bpmn.org/.

[11] Mary Lou Roberts and Paul D. Berger. *Direct Marketing Management*. Prentice Hall, 1999.

[12] Bastin Tony Roy Savarimuthu, Maryam Purvis, and Martin Purvis, “Different perspectives on modeling workflows in an agent based workflow management system,” In Rajiv Khosla, Robert J. Howlett, and Lakhmi C. Jain, editors, *Knowledge-Based Intelligent Information and Engineering Systems*, volume 3684 of Lecture Notes in Computer Science, pages 208-214. Springer Berlin / Heidelberg, 2005.

[13] J. W. Shepherdson, S. G. Thompson, and B. R. Odgers, “Decentralised workflows and software agents,” *BT Technology Journal*, 17(4):65-71, 1999.

[14] M. Stibor and M. Kutil, “Torsche scheduling toolbox: List scheduling,” In *7th International Scientific Technical Conference on PROCESS CONTROL*, Koutynad Desnou, Czech Republic, 2006.

[15] Bob Stone and Ron Jacobs. *Successful Direct Marketing Methods. Interactive, Database and Customer Marketing for the Multichannel Communication Age*. McGraw-Hill, New York, 2007.

[16] P. Sucha, M. Kutil, M. Sojka, and Z. Hanzálek, “Torsche scheduling toolbox for matlab,” In *IEEE Computer Aided Control Systems Design Symposium* (CACSD'06), pages 1181-1186, Munich, Germany, October 2006.

[17] Shuying Wang, Weiming Shen, and Qi Hao, “An agent-based web service workflow model for inter-enterprise collaboration,” *Expert Systems with Applications*, 31(4):787-799, 2006.

[18] WfMC. *The workflow reference model*. WfMC specification documents, Workflow Management Coalition, 1994. Document Number: TC00-1003.

[19] WfMC. *XPDL - XML process definition language*. WfMC specification documents, Workflow Management Coalition, 2005.

[20] Yuhong Yan, Torsten Kuphal, and Jürgen Bode, “Application of multi-agent systems in project management,” *International Journal of Production Economics*, 68(2):185-197, 2000.

[21] He Yanli, Yang Haicheng, He Weiping, Zhang Wei, and He Xinping, “Flexible workflow driven job shop manufacturing execution and automation based on multi-agent system,” In *IEEE/WIC/ACM International Conference on Intelligent Agent Technology* (IAT '06), pages 695-699, Hong Kong, China, 2006.