Using multi-coordination for the design of mobile agent interactions

Giancarlo Fortino, Alfredo Garro, Samuele Mascillaro, and Wilma Russo

***Abstract*— This paper proposes a *multi-coordination* approach for the design of mobile agent interactions. The approach is founded on the *multi-coordination* concept, which is a synergic exploitation of multiple coordination models which best fit interaction requirements. In particular, the proposed approach is based on two steps: (i) candidate design solutions are defined through a procedure which allows to identify the most effective coordination models for a given mobile agent interaction scenario; (ii) the defined candidate design solutions are quantitatively evaluated through a discrete-event simulation framework which allows for an easy evaluation of mobile agent interaction scenarios in terms of ad-hoc defined performance indices.**

Index Terms— **Agent Interaction Design, Mobile Agents, Multi-Coordination, Performance Evaluation.**

I. Introduction

Code mobility paradigms have been introduced to support the design and the implementation of flexible, dynamic and reconfigurable distributed applications in terms of software components which are not confined in a single run-time context for their entire lifecycle but can migrate autonomously or on-demand across different contexts [1]. Among them, the most fascinating paradigm is represented by the mobile agents, executing software components capable of autonomous migration by retaining code, data and execution state. Although it is advocated that the exploitation of mobile agents can provide many benefits [2], they have introduced specific and not yet fully addressed issues that actually limit their advertised wide-spread use [3]. An interesting issue concerning with the design of mobile agent interactions regards how to clearly identify which agents will be interacting and how their interactions can be modeled. To deal with mobile agent interactions, communication paradigms and mechanisms as well as coordination models and architectures for non mobile software components have been enhanced to be mobility-aware (message-passing, tuple space, publish/subscribe, etc) and new ones have been purposely defined for logical and physical mobility (meeting, blackboard, shared transiently tuple spaces, reactive tuples) [4, 5, 6, 7, 8, 9]. Although some mobile agent frameworks already offer several mechanisms based on the aforementioned communication/coordination paradigms and architectures, in the current practice mobile agent interactions are designed on the basis of a single paradigm which is mainly based on message passing or, in some application domains, on tuple spaces [4]. As single model based communication/coordination might not be effective for satisfying all needs of mobile agent interactions in all possible application scenarios, the exploitation of multiple communication/coordination paradigms, namely *Multi-Coordination*, can enhance design effectiveness, improve efficiency, and enable adaptability in dynamic and heterogeneous computing environments [10]. In particular, *Multi-Coordination* allows agents to choose among a variety of different communication/coordination paradigms which best fit mobile agent interaction needs. Moreover, although several design patterns have been proposed for driving the design of mobile agent interactions [11, 12] and programmable coordination models and related frameworks (e.g. TuCSoN [13]) are now available, systematic methods for supporting the development of mobile agent interactions which specifically take into account an integrated exploitation of multiple coordination models are surprisingly still lacking. To overcome this lack, this paper proposes a *multi-coordination* approach for the *design* and *evaluation* of mobile agent interactions.

The *design* is based on a procedure which uses suitable agent interaction patterns to fulfill agent coordination requirements. In particular, interaction patterns are first characterized by appositely defined parameters and associated to specific coordination models according to such parameters; then, the most appropriate coordination model is selected for implementing a given interaction pattern so providing a design solution for the related coordination requirement.

The *evaluation* is based on a *discrete-event simulation framework* which allows to evaluate the designed solutions in

G. Fortino is with the Department of Electronics, Informatics and Systems (DEIS), University of Calabria, Rende (CS), 87036 Italy. (corresponding author; phone: +39.0984.494063; fax: +39.0984.494713; e-mail: g.fortino@unical.it).

A. Garro is with the Department of Electronics, Informatics and Systems (DEIS), University of Calabria, Rende (CS), 87036 Italy. (e-mail: garro @unical.it).

S. Mascillaro is with the Department of Electronics, Informatics and Systems (DEIS), University of Calabria, Rende (CS), 87036 Italy. (e-mail: samuele.mascillaro@deis.unical.it).

W. Russo is with the Department of Electronics, Informatics and Systems (DEIS), University of Calabria, Rende (CS), 87036 Italy. (e-mail: w.russo @unical.it).

terms of performance indices with reference to given application scenarios. In particular, the *simulation framework* provides effective abstractions for easily programming mobile agent interaction scenarios and flexibly supporting configuration, execution and evaluation of such scenarios.

The proposed *multi-coordination* approach makes it possible the definition of alternative design solutions and their evaluation and comparison from qualitative (i.e. according to design effectiveness criteria) and quantitative (i.e. according to performance indices) points of view.

To show a concrete application of the proposed approach, a significant case study related to mobile agent-based distributed information retrieval is presented. In particular, some design solutions, which use different coordination models (message-passing, Linda-like tuple space, publish/subscribe), are defined on the basis of specific agent coordination requirements. Among the designed solutions, *multi-coordination-based* and *message-passing-based* solutions have been evaluated against significant performance indices. The evaluation shows that the *multi-coordination-based* solution has the best overall performance.

The remainder of this paper is organized as follows. Section II provides some background concepts about mobile agent coordination and discusses related work. By using a case study Section III exemplifies the design of alternative solutions through a three-step procedure. Section IV briefly proposes a comparison of the results of the performance evaluation of two alternative solutions based on multi-coordination and message-passing. Finally conclusions are drawn and on-going work is briefly elucidated.

II. COORDINATION AMONG MOBILE SOFTWARE AGENTS

Coordination basically implies the definition of a coordination model and related coordination architecture or related coordination language. In particular, in the context of Agents, an agent coordination model [14] is a conceptual framework which should cover the issues of creation and destruction of agents, communications among agents, and spatial distribution of agents, as well as synchronization and distribution of their actions over time. In this framework, the *coordinables* are the coordinated entities (or agents) whose mutual interaction is ruled by the model, the *coordination media* are the abstractions enabling the interaction among the agents, and the *coordination laws* are the rules governing the interaction among agents through the coordination media as well as the behavior of the coordination media itself. To date, agent coordination models have been classified by using several taxonomies [4**,** 15]; for example they can be classified in control-driven and data-driven according to the taxonomy proposed in [15]. However, in this paper the reference taxonomy is that proposed in [4] as the focus is on agents strongly characterized by mobility. It is worth noting that, although mobility can be an enabling feature for improving efficiency and effectiveness in distributed systems, mobility poses further issues on agent coordination as mobile entities demand for more complex coordination frameworks. The reference taxonomy [4] for Internet-based mobile agent coordination takes these issues into consideration and, in particular, classifies coordination models on the basis of the degrees of spatial and temporal coupling induced by the coordination models themselves. *Spatial coupling* requires that the entities to be coordinated share a common name space or, at least, know the identity of their interaction partners; conversely, *spatial decoupling* allows for anonymous interaction, i.e. there is no need for an acquaintance relationship. *Temporal coupling* implies synchronization of the interacting entities whereas *temporal decoupling* allows for asynchronous interactions [4].

On the basis of the reference taxonomy the following coordination models have been classified: Direct, Meeting-oriented, Blackboard-based and Linda-like.

In *Direct* coordination models, agents usually coordinate using RPC-like primitives or asynchronous message passing. The former coordination method implies temporal and spatial coupling whereas the latter implies only spatial coupling as temporal decoupling can be obtained by adopting message reception queues [16]. The majority of the Java-based mobile agent systems [17], particularly the most famous ones, namely Aglets, Voyager, Ajanta and Grasshoppers, rely on this model.

In *Meeting-oriented* models, agents coordinate using implicit or known meeting points which allow for partial spatial decoupling.

In *Blackboard-based* models, agents coordinate via shared data spaces to store and retrieve information under the form of messages so providing only temporal decoupling.

In *Linda-like* models, agents coordinate through tuple spaces which allow for insertion and retrieval of tuples by using associative pattern-matching; this enables both spatial and temporal decoupling.

Recently new coordination models which can be classified as spatially and temporally decoupled have emerged in the context of Internet applications: the *reactive tuple space* models which enable programmable coordination spaces [18, 19], *transiently shared tuple space* models which handle interactions in the presence of active mobile entities [20], and the *publish/subscribe event-based* models [6**,** 21**,** 22].

The *reactive tuple space* model extends the simple tuple space model by introducing computational capability inside the coordination media under the form of programmable reactions, triggered by operations on the tuple space or by other reactions, which can influence the behavior of agents. This model also allows for the separation of concerns between agent computation and coordination issues.

The *transiently shared tuple space* [20] is another Linda-like coordination model. As Linda offers a static, persistent and globally accessible tuple space, which is scarcely usable in presence of (physical or logical) mobility, the transiently shared tuple space model attempts to deal with these issues. In particular, each mobile agent owns a personal tuple space, named ITS (Interface Tuple Space). Whenever a mobile agent migrates, its ITS is carried with it and merged to the other co-

located agent's ITS making a transiently shared tuple space. Shared means that co-located agents can interact through the merged tuple space and transient means that its contents changes according to agent migrations.

In the *Publish/Subscribe event-based model*, agents coordinate through asynchronous publication and notification of events so enabling temporal and spatial decoupling [6]. In particular, to be notified about a published event an agent has to previously subscribe to the topic/type/context of the published event.

Each of the aforementioned coordination models has some features which make them suitable in given interaction patterns but poorly efficient or not usable at all in other patterns [15]. In [23] the authors proposed the use of multi-paradigm to design heterogeneous applications through different programming paradigms. On the basis of the multi-paradigm approach, a multi-coordination model [10] for the design and implementation of coordination among mobile agents executing in heterogeneous and dynamic distributed systems has been proposed.

III. A MOTIVATING EXAMPLE FOR THE MULTI-COORDINATION

This section proposes a simple yet effective case study which motivates the exploitation of multi-coordination for improving design effectiveness and, notably, system performances. The defined case study concerns with a distributed information retrieval task in a distributed computing system which is carried out through a coordinated set (or task force) of mobile agents. In particular, a user can search for specific information over a network of federated information locations by creating and launching a task force of mobile agents (called *searcher agents*) onto different locations. As soon as the task force finds the desired information, the user is notified with the found information. The proposed solution for the coordination of the task force during its information retrieval task implies that the following coordination requirements (CRs) are to be fulfilled:

- CR_1: every time a *searcher agent* visits a location not yet searched by other agents of the same task force, it notifies the other agents that such location has already been searched so avoiding unnecessary and resource-consuming duplicate searches.
- CR_2: as soon as a searcher agent finds the desired information on a given location, it reports the found information to the user.
- CR_3: when a searcher agent finds the desired information on a given location, it signals such event to all the other searcher agents to stop them.

These coordination requirements (CR_1, CR_2, CR_3) can be respectively designed by the following commonly used mobile agent interaction patterns (LBN, R2O, GBN) [4, 11, 12]:

- *Location-based notification* (LBN), which involves agents passing through a given location to be notified about events occurring/occurred in such location.
- *Report to owner* (R2O), which involves a child agent reporting to its owner agent when its task is completed.
- *Group-based notification* (GBN), which involves an agent notifying all its peer agents when a given event occurs.

These interaction patterns must be effectively implemented by exploiting the most appropriate coordination model/s which can be identified through the following subsequent steps:

1. *Characterization* of the interaction patterns according to appositely defined parameters by taking into account some application-level constraints;
2. *Matching* of the characteristics of the interaction patterns with the intrinsic features of the considered coordination models.
3. *Selection* of the most appropriate coordination model according to specific criteria using the results of the *Matching* step.

The defined parameters for the *Characterization* step of mobile agent interactions are:

- *Number of participants (PN)*, which can assume values in the range [2..N].
- *Participant identity (PI)*, which concerns with the mutual knowledge among interacting agents. PI can therefore assume the values *known* or *unknown*.
- *Locus (L)*, which indicates remote or local interactions among agents. L can assume the values *local* or *remote*.
- *Temporality (T)*, which refers to the type of temporal coupling among interacting agents. T can assume two values: *async* for time decoupling and *sync* for time coupling.

The characterization of the considered interaction patterns is reported in Table 1 in which the PI characteristic of the LBN and GBN cannot be fixed as the agents of a task force may or may not know the identity of each other

TABLE 1. CHARACTERIZATION OF THE INTERACTION PATTERNS

INTERACTION PATTERN	DIMENSIONS			
	PN	*PI*	*L*	*T*
LBN	2..N	UNKNOWN / KNOW	LOCAL	ASYNC
R2O	2	KNOWN	REMOTE	ASYNC
GBN	2..N	UNKNOWN / KNOW	REMOTE	ASYNC

To carry out the *Matching* step, it is needed to characterize the considered coordination models with respect to the characteristics of the interaction patterns to identify what characteristics they are able to intrinsically support. In particular the considered coordination models are the following:

- *Queue–based unicast asynchronous message passing* (QUAMP), which supports a variable number of participants, allows for both local and remote interactions and does not require temporal coupling between participants.
- *Local Linda-like tuple space* (LTS), which supports a high number of participants, allows temporal decoupling but only local interaction is supported.
- *Topic-based publish/subscribe* (TPS), which supports a high number of participants, allows for both local and

remote interactions and does not require temporal coupling between participants.

The *Matching* step intersects the characteristics of the defined interaction patterns with the characteristics supported by the considered coordination models to identify which coordination model is more suitable to implement a given interaction pattern. As the PI characteristic of the LBN and GBN depends on mutual knowledge among agents (the considered application-level constraint), the *Matching* step produces two possible matchings, reported in Tables 2 and 3, which are respectively related to the value assumed by the PI characteristic (*unknown* or *known*).

TABLE 2. CHARACTERISTICS OF INTERACTION PATTERNS WHICH CAN BE DIRECTLY SUPPORTED BY A CORDINATION MODEL (PI=UNKNOWN FOR LBN AND GBN)

IP	CM	Characteristics			
		PN	PI	L	T
LBN	**LTS**	**X**	**X**	**X**	**X**
	TPS	X	X		X
	QUAMP			X	X
R2O	LTS	X			X
	TPS	X		X	X
	QUAMP	**X**	**X**	**X**	**X**
GBN	LTS	X	X		X
	TPS	**X**	**X**	**X**	**X**
	QUAMP			X	X

TABLE 3. CHARACTERISTICS OF INTERACTION PATTERNS WHICH CAN BE DIRECTLY SUPPORTED BY A CORDINATION MODEL (PI=KNOWN FOR LBN AND GBN)

IP	CM	Characteristics			
		PN	PI	L	T
LBN	**LTS**	**X**	**X**	**X**	**X**
	TPS	X	X		X
	QUAMP	**X**	**X**	**X**	**X**
R2O	LTS	X			X
	TPS	X		X	X
	QUAMP	**X**	**X**	**X**	**X**
GBN	LTS	X	X		X
	TPS	**X**	**X**	**X**	**X**
	QUAMP	**X**	**X**	**X**	**X**

The *Selection* step, which allows to choose the coordination model which best supports the characteristics of an interaction pattern, is based on the following selection criterion: the coordination models supporting all the characteristics of an interaction pattern will be the candidate models to implement such interaction pattern.

TABLE 4. DESIGN SPACE FOR PI=UNKNOWN

IP	CM	Implementation description
LBN	LTS	When a searcher agent searches in a location which has not been already searched by another agent of its task force, it inserts (by using the *out* primitive) a signaling tuple into the LTS to signal that this location has been searched. As soon as an agent visits a location and reads the signaling tuple (by using the *non-blocking rd* primitive), it avoids searching.
R2O	QUAMP	When a searcher agent finds the desired information, it sends a message containing the found information (by using the *send* primitive) to its owner.
GBN	TPS	When a searcher agent finds the desired information, it publishes an event of a specific topic related to its task force (by using the *publish* primitive) which signals the stop of the retrieval task. All the other agents of the task force will be thus asynchronously notified since they subscribed to the specific topic at creation time.

According to such criterion the only possible solution if PI=*Unknown* (see Table 2) is represented by the following coordination models: LTS for LBN, QUAMP for R2O and TPS for GBN. An implementation of such solution is reported in Table 4 which constitutes the related *design space*.

Conversely, if PI=*Known* (see Table 3), the coordination models which can be selected are LTS and QUAMP for LBN, QUAMP for R2O, and QUAMP and PS for GBN. The related *design space* which contains the implementations of the interaction patterns through the selected coordination models is reported in Table 5.

TABLE 5. DESIGN SPACE FOR PI=KNOWN

IP	CM	Implementation description
LBN	LTS	**see table 4**
	QUAMP	A searcher agent to notify that it has searched a given location sends a message containing the location identifier (by using the *send* primitive) to all the other searcher agents of the task force.
R2O	QUAMP	**see table 4**
GBN	TPS	**see table 4**
	QUAMP	A searcher agent which has found the desired information sends a notification message (by using the *send* primitive) to all the other searcher agents of the task force to stop them.

The choice of a specific solution among alternative solutions (if any) can depend on different criteria bounded to the values which can be assumed by specific characteristics of the interaction patterns. In particular, this choice can be driven by qualitative considerations or by performance evaluation of the alternative design solutions.

With reference to Table 5, the following qualitative considerations based on the values of the PI characteristics can be considered:

- if the number of participants is very large (PI>>2), the GBN interaction pattern could be better supported by TPS as an agent to notify all the others through TPS always needs to send just one notification whereas the same notification based on QUAMP needs the generation of as many messages as the number of the participants. Thus the use of QUAMP leads to a bottleneck at the agent location both for the agent execution and network performances.
- if the number of participants is small, QUAMP could be a more effective choice as TPS requires a distributed middleware-level infrastructure more complex than that required by QUAMP.

The abovementioned considerations also hold for the LBN interaction pattern.

An example of performance evaluation for driving the choice among alternative design solutions is shown in the next section in which the evaluation and comparison of two possible design solutions based on *multi-coordination* and *message-passing* is presented.

IV. A PERFORMANCE EVALUATION OF THE DESIGNED SOLUTIONS: MULTI-COORDINATION VS. MESSAGE-PASSING

The proposed multi-coordination approach uses a *discrete-event simulation framework* for the evaluation of the designed solutions. The simulation framework provides effective *multi-coordination*-based programming abstractions [24] for the implementation of agent-based systems. In particular, the simulation framework is an enhancement of MASSIMO [25, 26] to support multiple coordination spaces through which agents can interact and currently includes an implementation

of the following coordination spaces:

- The asynchronous Message-based coordination space which is based on proxies [16]. In particular, a message is delivered at the agent home location and, from here, forwarded to the actual agent location by following the chain of proxies left during agent migration.
- The Publish/Subscribe coordination space which behaves like a state-full ELVIN event notification system [6]. In particular, before agent migration the system removes all existing subscription of the migrating agent and re-subscribes the agent to the same notifications after the agent arrives at the new location. Moreover the weight of a notification is less than the weight of a message as no source field of the notification is included.
- The Tuple coordination space which is based on TuCSoN [13]. In particular, each location has its own local tuple space, an instance of a TuCSoN tuple space which relies on text-based tuples.

According to the simulation framework two alternative solutions designed in section III (see Table 5), <LTS, QUAMP, TPS> (or *multi-coordination-based* solution) and <QUAMP, QUAMP, QUAMP> (or *message-passing-based* solution), have been implemented and simulated to calculate the ad-hoc defined performance indices reported in Table 6.

TABLE 6. EVALUATION PERFORMANCE INDICES

Name	Definition
T_{TC}	*Task completion time*: the temporal gap between the spawning of the first created Searcher Agent and the first report message received from the User Agent.
T_N	*Notification time*: the temporal gap between the information finding and the notification to the last Searcher Agent.
N_V	*Number of visits after finding the information*: the total number of locations visited by the Searcher Agents after the information finding.
N_S	*Number of searches after finding the information*: the total number of the locations searched by the Searcher Agents after the information finding.
N_M	*Number of coordination messages*: the number of coordination messages transmitted through the network.

The simulation tests rely on the simulation parameters (the number of locations and the number of *searcher agents*) and on the following settings of the simulation topology at network and information level:

- locations are connected through a fully connected logical network composed of FIFO channels. In particular, channels are characterized by the same delay and bandwidth parameters modeled as uniform random variables.
- the information to be found is contained exactly at one location and the locations keep references (randomly generated) to other locations at information level to be all reachable.

Simulations were carried out with the number of locations equals to 100 and the number of *searcher agents* in the range [10..90, step=10]. Moreover, for each simulation run, the *multi-coordination-based* and *message-passing-based* solutions were executed on the same network and information topologies. In Figures 1-5 the simulation results are reported; the obtained values of the performance indices were averaged over 100 simulation runs.

The T_{TC} performance index, which measures the speed with which the information search task is carried out, decreases as the number of searcher agents increases (see Figure 1). In fact, the use of more searcher agents augments the degree of parallelism which, consequently, increases the probability to find the searched information with a smaller number of migrations which are time-consuming. The performances of the *message-passing-based* and *multi-coordination-based* solutions are almost the same.

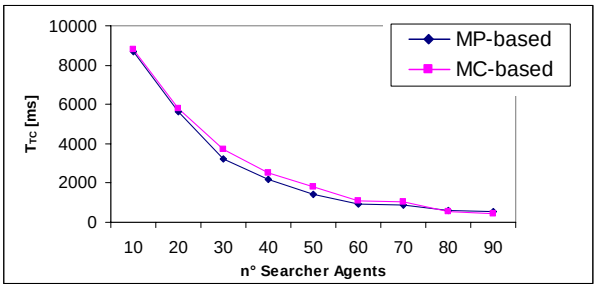

Figure 1: Task completion time

The T_N performance index measures how fast all the *searcher agents* are notified after finding the information. The shorter T_N, the fewer are the resources consumed throughout the networked agent platform. The *multi-coordination-based* solution performs better than the *message-based-solution* when the number of searcher agents is less than 80 (see Figure 2) due to (i) the exploitation of the TPS coordination space which provides faster notifications than the message-based coordination space and (ii) the network load which is lighter than the one obtained in the *message-passing-based* solution (see discussion about the N_M parameter). However, when the number of agents is greater than 80 the *message-passing-based* solution performs better as it avoids the occurrence of many migrations which could slow down the *stop* notification of agents. In fact, when the LBN interaction pattern is carried out through LTS, agents should migrate to a location to understand if such location has been searched. Conversely, when the LBN interaction pattern is carried out through QUAMP, agents send messages to notify a searched location to the others so limiting the number of migrations per agent as the agents are notified without having to migrate to new locations.

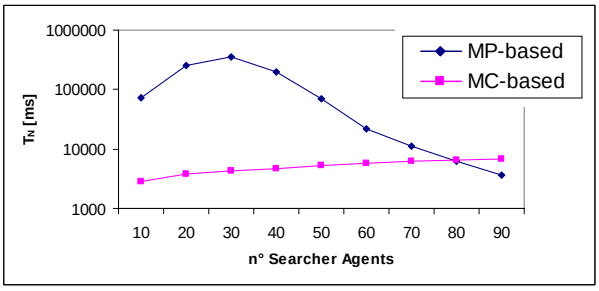

Figure 2: The notification time.

The N_V and N_S parameters are measures of the consumption of resources after the information is found. The values of such parameters should be kept as low as possible. As shown in Figures 3 and 4, the *multi-coordination-based* solution outperforms the *message-passing-based* solution when the number of searcher agents is less than or equal to 40.

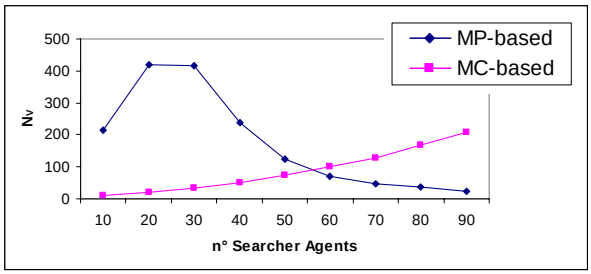

Figure 3: Number of visits after finding information

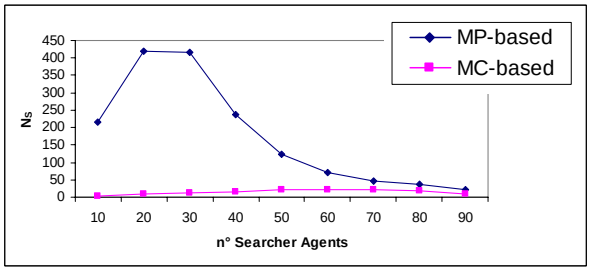

Figure 4: Number of searches after finding information

Finally the N_M parameter (see Figure 5), which measures the network load, is significantly better in the *multi-coordination-based* solution thus saving network resources with respect to the *message-passing-based* solution.

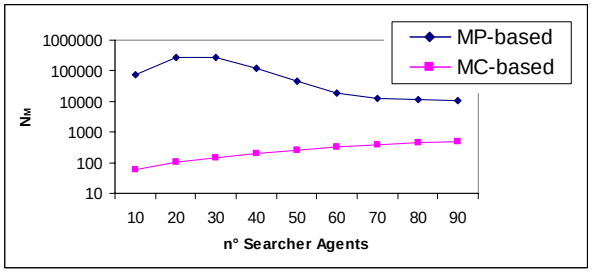

Figure 5: Number of coordination messages

Finally, it is worth noting that a network of locations cannot be flooded by a lot of agents per searching task which would cause an over usage of network resources, even though a numerous task force of agents would significantly decrease the T_{TC} as shown in Figure 1. So a trade-off should be reached in terms of the number of agents constituting the task force which should be appositely set to a percentage of the number of available locations to minimize the resource usage and obtain low task completion times. According to the obtained results (see Figures 1-5) this percentage can be set to 40% for the *multi-coordination-based* solution which is a good trade-off and implies that a task force of 40 agents is created and launched for each information retrieval task.

V. Conclusion

This paper has proposed a multi-coordination approach for the design and evaluation of mobile agent interactions which is based on two subsequent phases: (i) the defined coordination requirements among agents are designed through well-known agent interaction patterns which are then implemented by using specific coordination models according to a three-step procedure which provides alternative design solutions; (ii) these alternative design solutions are evaluated and compared through an agent-oriented discrete-event simulation framework according to ad-hoc defined parameters.

The proposed approach has been applied to a simple yet effective case study which has highlighted its actual applicability and that the exploitation of *multi-coordination* could be both more effective and more efficient than the use of a message-based coordination model.

On the basis of the obtained results work is underway for: (i) testing the proposed three step technique with a wide variety of coordination requirements, agent interaction patterns and coordination models; (ii) relaxing the selection criterion proposed in section III to also consider other solutions which can be implemented by mixing a coordination model with mobility and third-party agent components (e.g. reflectors, mediator, facilitator, etc); (iii) enhancing the simulation framework to include other coordination spaces.

References

[1] A. Fuggetta, G.P. Picco, and G. Vigna, “Understanding Code Mobility”, *IEEE Trans. on Software Engineering*, 24(5), pp. 342-361, 1998.

[2] D.B. Lange and M. Oshima, “Seven good reasons for Mobile Agents”, *Communications of the ACM*, 42, 3, pp 88-89, 1999.

[3] G. Vigna, “Mobile Agents: Ten Reasons For Failure”, *Proceedings of the 2004 IEEE International Conference on Mobile Data Management* (MDM’04), Berkeley, CA, USA, 19-22 January 2004.

[4] G. Cabri, L. Leonardi and F. Zambonelli, “Mobile-agent coordination models for internet applications”, *IEEE Computer*, *33*, *2*, pp 82-89, 2000.

[5] A. Murphy, G. P. Picco, and G. Roman, “LIME: A Middleware for Logical and Physical Mobility”, *Proceeding of 21th International Conference on Distributed Computing Systems*”, IEEE CS, 2001.

[6] A. Padovitz, “Agent communication using Publish-Subscribe genre: Architecture, Mobility, Scalability and Applications”, *Annals of Mathematics, Computing and Teleinformatics*, 1, 3, pp 35-50, 2004.

[7] J. Baumann, F. Hohl, N. Radouniklis, K. Rothermel and M. Strasser, “Communication concepts for Mobile Agent Systems”, *Proceeding of the 1st International Workshop on Mobile Agents (MA’97)*, Berlin, Germany, LNCS 1219, pp. 123-135, April 1997.

[8] S. Choi, H. Kim, E. Byun, C. Hwang, and M. Baik, "Reliable Asynchronous Message Delivery for Mobile Agents", In *IEEE Internet Computing*, vol. 10, no. 6, pp. 16-25, 2006.

[9] J. Cao, X. Feng and S.K. Das, "Mailbox-Based Scheme for Mobile Agent Communications", *Computer 35(9)*, pp. 54–60, 2002.

[10] G. Fortino and W. Russo, “Multi-coordination of Mobile Agents: a Model and a Component-based Architecture”, *Proceedings of 20th Annual ACM Symposium on Applied Computing (SAC’05), Special Track on Coordination Models, Languages and Applications*, Santa Fe, NM, USA, Mar. 13-17, 2005.

[11] Y. Aridor, D.B. Lange, “Agent Design Patterns: Elements of Agent Application Design”, *Proceedings of Autonomous Agent ’98*, Minneapolis, Minnesota, US, 1998.

[12] D. Deugo, M. Weiss, and E. Kendall, “Reusable Patterns for Agent Coordination” published as Chapter 14 in the book: Omicini, A.,

Zambonelli, F., Klusch, M., and Tolksdorf, R. (eds.), Coordination of Internet Agents: Models, Technologies, and Applications, Springer, 2001.

[13] A.Omicini and F. Zambonelli, "Coordination of Mobile Agents for Information Systems: the TuCSoN Model", *Proceeding of 6th AI*IA Convention*, 1998.

[14] P. Ciancarini, "Coordination models and languages as software integrators", *ACM Computing Surveys*, 28, 2, pp 300-302, Jun 1996.

[15] G.A. Papadoupolos, F. Arbab, "Coordination models and languages", In *Advances in Computers 46*, Academic Press, 1998.

[16] X.Y. Zhou, N. Arnason and S.A. Ehikioya, "A proxy-based communication protocol for mobile agents: protocols and performance", *IEEE Conference on Cybernetics and Intelligent Systems*, volume 1, pp 53-58, 1-3, Dec. 2004.

[17] A.R. Silva, A. Romao, D. Deugo and M. Mira da Silva, "Towards a reference model for surveying mobile agent systems", *Autonomous Agent and Multi-Agent Systems*, 4 (3), pp 187-231, 2001.

[18] G. Cabri, L. Leonardi and F. Zambonelli, "Engineering Mobile Agent Applications via Context-dependent Coordination", *IEEE Transactions on Software Engineering, 28, 11*, pp 1040-1056, Nov. 2002.

[19] A. Omicini, and F. Zambonelli, "Tuple centres for the coordination of internet agents", *Proceedings of ACM Symp. on Applied Computing (SAC'99), Special track on Coordination Models, Languages and Applications*, San Antonio, TX, USA, Feb 28-Mar 2, 1999.

[20] G. P. Picco, A. L. Murphy and G. C. Roman, "LIME: Linda meets mobility", 1999

[21] G. Cugola, E. Di Nitto and A. Fuggetta, "The Jedi event-based infrastructure and its application to the development of the OPSS WFMS", *IEEE Transactions on Software Engineering, 27, 9*, pp 827-850, 2001.

[22] A. Carzaniga, D.S. Rosenblum and A. Wolf, "Design and evaluation of a wide-area event notification service", *ACM Transactions on Computer Systems, 19, 3*, pp 332-383, 2001.

[23] P. Zave, "A compositional approach to MultiParadigm Programming", *IEEE Software 6(5)*, pp 15-25, 1989.

[24] G.Fortino, A. Garro, S. Mascillaro and W. Russo, "Modeling Multi-Agent Systems through Event-driven Lightweight DSC-based Agents", *Proceedings of 6th International Workshop From Agent Theory to Agent Implementation (AT2AI'06)*, May 13, 2008, AAMAS 2008, Estoril, Portugal, EU.

[25] M. Cossentino, G. Fortino, A. Garro, S. Mascillaro, and W. Russo, "PASSIM: a simulation-based process for the development of multi-agent systems", *International. Journal on Agent-Oriented Software Engineering* 2(2), 132-170, 2008.

[26] G. Fortino, A. Garro, and W. Russo, "A Discrete-Event Simulation Framework for the Validation of Agent-based and Multi-Agent Systems", *Proceedings. of the Workshop on Objects and Agents (WOA'05)*, Camerino, Italy, Nov 14-16, 2005.