The Artificial Collective Engine Utilising Stigmergy (ACEUS) a Framework for Building Adaptive Software Systems

G.B. O'Reilly and E.M. Ehlers

Abstract—Many software systems are deemed failures due to the systems inability to rapidly adapt to its ever-changing domain needs. Designing and developing software solutions along the lines of a Complex Adaptive System (CAS) ensures the robustness and adaptability of the software solution. The utilisation of a CAS model in a software solution has the benefits of enhanced flexibility, adaptability, distribution and reusability. Such benefits can strengthen the ability of business software solutions to evolve and survive in the market place. Especially on the eve of the Semantic web. The Artificial Collective Engine Utilising Stigmergy (ACEUS) framework is described under the properties and mechanisms of a CAS. ACEUS forms a framework for building distributed business solutions for the Semantic web. The ACEUS framework is base upon and utilses the ant colony analogy. The taxonomy of the ACEUS framework is presented in terms of the layers and the utilization of stigmergy.

Index Terms—Complex adaptive systems, Stigmergy, Distributed systems, Semantic web.

1 INTRODUCTION

THE predicted Semantic web will cause an ever increasing demand for modern distributed systems that are robust, adaptable and even self-organizing. Web-accessible programs and a variety of other physical devices will be dependant on the services supplied by the semantic web. This ever increasing demand to access networked applications on the semantic web will cause a dramatic change in the scale, reliability, availability, adaptability, security and complexity of current distributed systems. The distributed systems on which the semantic web is to be built will need to be extremely dynamic, adaptive, robust and self-organizing. Characteristics not exhibited by traditional (current) distributed systems.

At present the traditional distributed systems are designed and developed in an inflexible and centralized manner based on client-server architectures. The most common motivation for the centralized, top-down design of current distributed systems is easier system management. These methods of distributed systems design are showing their vulnerability when dealing with adaptiveness, self-organization and an enormous amount complexity. With the event of the semantic web distributed systems will be reaching a point of complexity that will put them far beyond traditional or classical means of management and deployment. A means of constructing adaptable, robust, scalable, self-organizing and self-repairing distributed systems is needed.

- *G.B. O'Reilly is with the Department of Computer Science, Rand Afrikaans University, Auckland Park, Johannesburg, South Africa. E-mail: oreill_g@mtn.co.za.*
- *E.M. Ehlers is with the Department of Computer Science, Rand Afrikaans University, Auckland Park, Johannesburg, South Africa. E-mail: eme@na.rau.ac.za.*

Manuscript received (insert date of submission if desired). Please note that all acknowledgments should be placed at the end of the paper, before the bibliography.

Commercial organizations will be a major driving force behind the building and promotion of the semantic web. It is vital that business systems that utilise the semantic web as well as acting as services on the semantic web have the characteristics to confront the coming complexity explosion. A demand has arisen for a dynamically adaptive framework for create business systems that have the characteristcs to continually adapt to change and that are robust enough to cater for that change dynamically. Dynamically adaptive frameworks will accomodate change without having to be disassembled and redesigned.

The Artificial Collective Engine Utilising Stigmergy (ACEUS) framework attempts to cater for this type of dynamical adaptation and self organization. The ACEUS framework is an attempt to model an artificial complex adaptive system. The ACEUS framework is modeled around the ant colony metaphor. The ACEUS framework will be ideal for commercial organizations that what to integrate and utilise the semantic web in their daily business. ACEUS will also help automate and streamline business processes to adapt intelligently and optimally to changing business environments.

A Complex Adaptive System (CAS) has been defined as systems composed of interacting agents that are diverse in both form and capabilities [1]. The individual agent's behavior is governed by a collection of rules. These agents respond to stimuli and the stimulus-response exhibited is a result of the rules embedded in the agent [1]. Major improvements in adaptivity and robustness can be achieved in software solutions by integrating the typical properties and mechanisms of CAS defined by Holland [1], [2], [3].

Inter communication between the agents and between the agents and their environment is referred to as interactions. Interactions cannot be anticipated since the modifications induced in the environment by the agents'

actually influence the future interactions of these agents. Interactive systems can never be fully specified nor fully tested due to their undeterministic nature. Business systems that have been developed that incorporate interactive software agents can be categorized as complex adaptive systems. CAS is defined by Holland as a system composed of interacting agents that respond to stimuli and is characterised by stimulus-response behavior that can be defined in terms of rules [1], [2], [3]. The ACEUS framework takes advantage of the idea that the undeterministic interactions between agents and between agents and their environment may result in emergent behavior a fundamental feature of a CAS. The framework also utilises the idea of creating rules for the agents that allows them to respond to various stimuli in the environment. The response to stimuli in the environment allows the agents to interact with each other and with the environment. These interactions in the ACEUS framework are modeled on stigmergy taken from the ant colony analogy [4], [5].

2 ANT SYSTEMS

A new paradigm of evolutionary computation has been discovered with the ant colony metaphor. One of the most successful application models based on the ant colony analogy is the Ant Colony Optimization model (ACO) [6]. Another approach to the ant colony analogy is Ant System (AS) proposed by Dorigo, Maniezzo and Colorni [7] and the Ant Colony System (ACS), which was a refinement of the Ant System. The Ant system has shown promising results when applied to the traveling salesman problem (TSP) while the Ant Colony System has displayed competitive results with some of the best-known heuristics for the TSP [8]. Schoonderwoerd, Holland, Bruten and Rothkrantz [9] and Bonabeau, Henaux, Guerin, Snyers, Kruntz and Theraulaz [10] have also proposed ant algorithms for network routing and load balancing in telecommunication networks. The ant foraging algorithms have also been used for cargo operations such as cargo routing and handling systems on airlines [11]. Valckenaers proposed concepts for designing agent-based control systems for manufacturing environments using stigmergy [12]. According to this proposal stigmergy can be utilised to model suitable control behavior.

The ACEUS framework differs completely from previous ant algorithms and stigmergy control models. The ACEUS framework utilises the analogy of an ant colony to create a software engine that will be distributed, self-organizing, adaptable and robust. This will allow automatic configuration and optimization of a system in a complex environment. A complex environment can be seen as an environment that is non-linear, unpredictable and continually changing, for example the semantic web. The need for adaptability, intelligent self-reconfiguration and self-optimization is essential in a complex environment.

3 ACEUS FRAMEWORK

ACEUS as the name depicts utilises stigmergy as a communications mechanism where as other models have used stigmergy as a controlling mechanism [12]. In ACEUS the stigmergy layer does not control the system; it records changes in the environment induced by agents. These changes then act as stimuli to the agents in the collective. Due to the agents responding to the stimuli in the stigmergy layer, the stigmergy layer is actually influencing the behavior of the agents in the collective. Agents induce the changes in the stigmergy layer by depositing pheromones or objects in this layer. All agents in the ACEUS framework are self-contained and execute independent of one another. However, agents can indirectly influence one another by inducing changes in the stigmergy layer. The stigmergy layer simulates the environment. The environment acts as another member of the collective. When a pheromone or object is deposited in the stigmergy layer the actual changing of the environment in the stigmergy layer stimulates various agents. In a sense the environment in the stigmergy layer is acting as a driving force for stimulating responses from the agents. There is no central controlling mechanism in the stigmergy layer.

The ACEUS framework defines a collective that can be thought of as a cybernetic type collective as it incorporates both organic and *in silica* agents. Organic agents are agents composed of organic material such as human beings. *In silica* agents are silicon-based agents that are created by and exist on a computer system. Moreover, this cybernetic type collective is limited on the organic agent side since the organic agents can only interact with the collective via interfacing agents. Interfacing agents are also silicon-based agents. An example of an interfacing agent would be an agent with a user interface that is dependant on input from an organic agent. The organic agent is constrained to the functionality that the interfacing agent provides and has to operate to these constraints. Any interaction between an organic agent and an *in silica* agent will happen via the interfacing agent.

Since ACEUS is based on a complex adaptive system many individual agents exist in the framework. The ACEUS framework has two perspectives, namely: the collective's perspective and the individual's perspective. The collective's perspective is the perspective from an observer outside the collective who is able to comprehend and understand the entire collective. The individual's perspective is the perspective from an individual inside the collective i.e. an agent that cannot comprehend nor understand the entire collective. The ACEUS framework's ontology can be seen from these two perspectives. Firstly if the perspective were at the level of the individual agents in the collective the ontology would appear as small segments or atom ontologies. These segments of the ontology are accesable to specific individual agents. An individual agent can only comprehend its segment. It cannot comprehend the entire ontology of the collective as it only has access to its segment. However, if the perspective is changed and the ontology is viewed from the collective's perspective the entire ontology is seen. The collective or swarm entity has the comprehension of the entire ontology.

Similarly, the rule base of ACEUS can also be seen from

the two different perspectives. The intelligence in the ACEUS framework is embedded into the individual agents in the form of rules. Each agent has a rule set that constitutes the individual agent's intelligence. However, the individual agent cannot comprehend the intelligence of the collective it can only comprehend its own rule set. The ACEUS framework when viewed as a collective seems to be an entity that possesses the intelligence of the entire collective and maybe even more due to emergence [13], [14]. The ACEUS framework emerges as an intelligent entity just as an ant colony emerges from the collaboration of the individual ants.

4 LAYERS

The ACEUS framework has a layered design. The five layers are: the ontology layer, the rule base layer, the *in silica* agent layer, the stigmergy layer and the organic agent layer (see Fig. 1). The ACEUS framework layers allow the framework to be easily understood and accessed. These layers add to the robustness and flexibility of the ACEUS framework design.

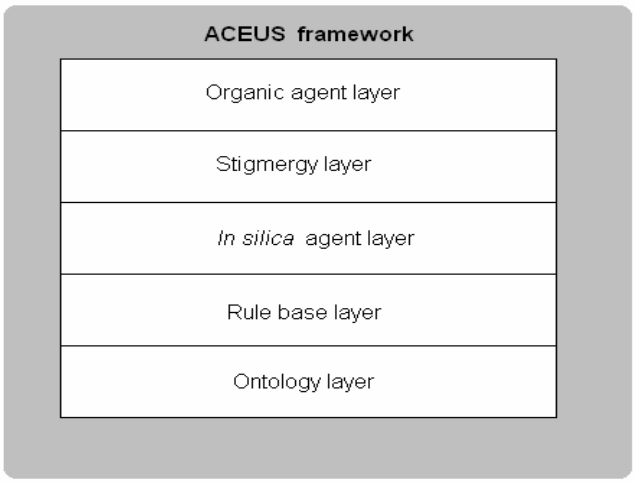

Fig. 1. The ACEUS framework layers.

4.1 Organic layer

Similar to the ants constituting the individual agents in the ant colony the organic and *in silica* agents are the individual agents in the ACEUS framework. Having a cybernetic type collective is very useful in a business type environment. When a situation arises where human intervention is needed i.e. a decision that a business organization would prefer a human to handle instead of an automated *in silica* agent (e.g. a morale decision), the decision can be handled by the organic layer of the framework. Thus the framework makes allowance for human intervention, but sees the intervention as another agent altering the environment (see Fig. 2).

Organic agents are able to request certain tasks or make recommendations in the cybernetic colony by depositing pheromones in the stigmergy layer via the interfacing agent (see Fig. 2). All agents in the ACEUS framework are unaware of any of the other agents in the framework this includes the organic and interfacing agents. The only interaction or communication between the agents is via the stigmergy layer. For example an organic agent will have to place a pheromone in the stigmergy layer via the interfacing agent to communicate with the inner layers as depicted in Fig.2.

Organic agents and interfacing agents are the main players in this layer. Organic agents in ACEUS are simply the human users of the system. The framework sees all human interaction in the framework as interactions of agents in the environment. The framework does not see the difference between an organic agent and an *in silica* agent. The ACEUS framework also does not differentiate between the organic agents, so no hierarchical structure can take place.

Thus all recommendations made to the system are considered. Recommendations that are advantageous to the system will be adopted while other recommendations that are disadvantageous will be rejected. The hierarchical structures formed in an organization do not influence the ACEUS framework at all. For example if a high ranking individual in the organisation makes a recommendation that is not advantageous to the system that recommendation will be rejected. Whereas recommendations that are advantageous to the system made by low ranking individual in the organization will be accepted. In short the ACEUS framework utilises it business intelligence to make decision that are advantageous to the collective i.e. the organization and not decisions that are based on the hierarchy of individuals.

The limitation of the organic agents is that the organic agent can only interact with the collective via an interfacing agent. Thus the organic agent is limited to the functionality offered by the interfacing agent. For example if an organic agent wants to initiate a request or recommendation in the system, the organic agent will use the interfacing agent to set that request up and the interfacing agent will drop the request in the form of a pheromone into the stigmergy layer (see Fig. 2). A pheromone in the ACEUS framework is a message object in the form of an XML file. The pheromone has properties that define the characteristics of the pheromone. Organic agents can deposit pheromones into the stigmergy layer via the interfacing agent.

The interfacing agent will contain all the traditional graphical user interfaces (GUI). GUIs are needed for the organic agent to interact with the interfacing agent. All data about the environment needed to create pheromones or objects will be given to the organic agent via the interfacing agent. Again due the advantage of the layers in the framework new interfacing agents can be dynamically added to the organic layer without affecting any of the other layers. Interfacing agents can vary from full-blown applications to small HTML pages.

4.2 Stigmergy layer

Stigmergy is the coordination of tasks and regulation of constructions (e.g. an ant hill in an ant colony and a business system in the ACEUS framework) in an environment that does not depend on the agents, but on the

constructions themselves [5]. The agent does not direct the work but is guided by it. In the ACEUS framework, stigmergy is defined as the influence the changing environment has on the agents in the environment. It is assumed that the constructions created by the agents in the environment form part of the environment. These constructions change the environment and the changing environment stimulates a certain response in the agents. In stigmergy the fundamental mechanism is the ability to use the environment as a shared medium for storing information so that other individuals can interpret it [4].

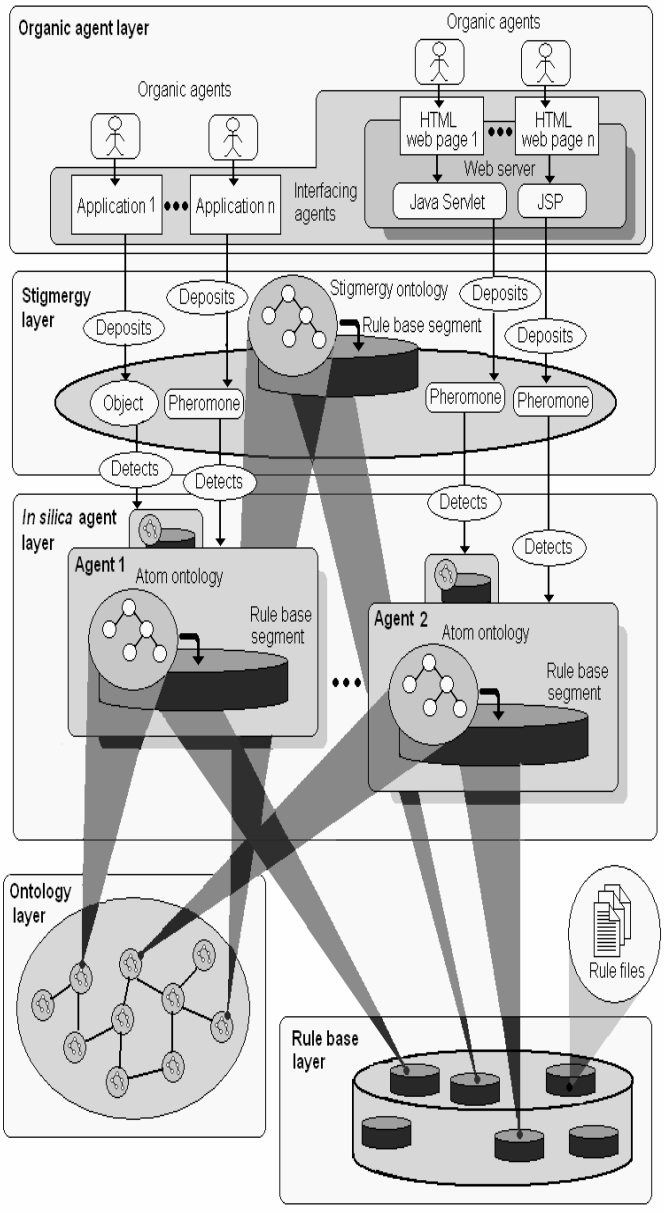

Fig. 2. The ACEUS framework depicted with its various layaers and components. The components of each layer and their interconnections are also depicted.

4.2.1 Stigmergy in natural systems

Pheromone signaling and object signal is demonstrated in natural systems such as ant and termite colonies. Object signaling can be described by a termite colony. Termites drop small pieces of mud randomly in the environment. These small pieces of mud can be seen as objects changing the environment. The change is signaled in the environment by the stigmergy layer when the object is moved or dropped. This can be referred to as object signaling. If several pieces of mud are randomly placed next to one another in the environment, the signaling of change in the environment at that specific point will be increased several times. Due to the increased signaling at that point the probability of the termite agents being stimulated enough so that they respond by dropping their mud objects at this location is increased. So long as the termite agent has a stimulus-response rule that allows a response of dropping its object at a point of increased signaling. The changing environment thus influences the termite agents on where to drop their mud objects. The larger the signaling the more attractive it is for further agents to drop their mud objects at this location. This results in small heaps been abandoned and larger heaps growing into tall columns.

To demonstrate pheromone signaling in ant colonies assume two trails A and B have been established between the nest and a food sources. Trail B is slightly shorter than trail A. Initially there will be the same amount of ant agents following trail A as there would be following trail B as the two trails cannot be differentiated initially. As the ants follow the trail they deposit pheromones in the environment. Again the stigmergy layer will begin signaling a change in the environment. Once an ant finds the food source its then starts making its way back to the nest depositing food pheromones in the environment. The ants following trail B will obviously return to the nest the quickest and then want to return to the food source. These ants will then follow the highest signaling in the environment pointing toward food i.e. the strongest scent from the pheromones. This would be trail B as the signaling along this trail would have been reinforced with food pheromones from the ant that have just returned to the nest from the food source. This would result in more pheromones been deposited on trail B and the signaling of change in the environment along trail B would continually increase. Thus the stimuli for the ant to follow trail B instead of trail A would continually increase. Eventually, because of this positive feedback, the longer trail A will be abandoned, while the shorter trial B will attract all the traffic. The ant agents are constantly tracing and updating an intricate network of trails in the environment that indicate the most efficient ways to reach different food sources. Individual ants do not need to keep the locations of the different sources in memory all they need is the stimulus response rules to follow the highest signaling trails in the environment. Thus a collectively developed trail network emerges that will always be there to guide them.

The two examples of pheromone signaling and object signaling may seem similar. However, the difference is that the ants leaving pheromone are not making any physical contribution to the solution of their problem (collecting food), unlike the termites whose actions directly contribute

to the mound building. With pheromone signaling the agents are merely providing the collective with a map to guide them through the environment. In fact, the trail network functions like an external mental map, which is used and updated by all the agents. This is a typical example of a collective memory map.

4.2.2 Stigmergy in the ACEUS framework

In the ACEUS framework the environment in which the agents deposit and detect their pheromones and objects is incorporated into the stigmergy layer i.e. the stigmergy layer simulates the environment. Any changes to the environment are signaled in the stigmergy layer. Thus if the ACEUS framework was simulating an ant colony, then the stigmergy layer would represent the actual spatial environment in which the ants roam, deposit and detect pheromones. Similarly, when the ACEUS framework is applied to a commercial organization, the stigmergy layer must represent the environment in which the organization operates.

The stigmergy layer in ACEUS signals any changes in the environment. The changes to the environment induced by the deposited pheromones or objects stimulate the agents to respond. The stigmergy layer is the communication layer of the ACEUS framework. Agents inter communicate in the ACEUS framework indirectly by depositing pheromones or objects in the environment. The changing environment communicates with the agents by acting as a hidden driving force that influences the agent's actions. Thus due the critical influence the environment plays in communication both between agents and between agents and the environment the stigmergy layer is the ideal layer for simulating the environment.

When simulating an environment in a computer model the conceptual environment could be captured in a collective ontology. A collective ontology is a huge ontology made up of a number of smaller ontologies called Atom ontologies. An Atom ontology is designed to capture the conceptual intelligence of a single agent in the ACEUS framework. Thus no agent will be able to comprehend the collective ontology it will only be able to comprehend its own ontology. Agents cannot utilise the collective ontology they can only utilise their own atom ontology. Any violation of this rule would severely hamper the idea of emergent behavior, as the size of the controlled content of the collective ontology would not be able to grow beyond the cognitive capabilities of an individual agent. To ensure that no problems of this nature occurs in the ACEUS framework agents will only be able to deposit pheromones or objects in the stigmergy layer. They will not be able to comprehend the stigmergy layer or understand the mechanisms of the stigmergy layer.

The messaging mechanism in the stigmergy layer can be seen in Fig. 3. The stigmergy layer utilises an embedded rules engine. When an agent deposits a pheromone or object into the assertion pool of the stigmergy layer a rule will fire depending on the pheromone or object type. This rule will then stimulate an existing agent by setting its detecting parameter to true. The rule will also retract the pheromone or object from the rules engine. Once an agent's detecting parameter has been set to true it will respond to this stimuli by performing its dedicated task. After the agent has completed its tasks it will set it's detect parameter too false. In this fashion the stigmergy layer is acting as a stimuli provider to the agents in the ACEUS framework. It also ensures a driving force for stimulating responses in the agents. The agents have no control on the stigmergy layer and cannot comprehend the means by which they are being stimulated. The main controlling mechanism in the stigmergy layer is the rules engine. The rules in the stigmergy layer are able to stimulate the agents into performing certain tasks. This is a modeled replica of the changing environment influencing agents such as ants and termites to perform certain tasks. The rules in the stigmergy are defined in the rule base layer (see Fig. 2). All the different pheromones and objects that can be deposited in the stigmergy layer are defined in the ontology layer.

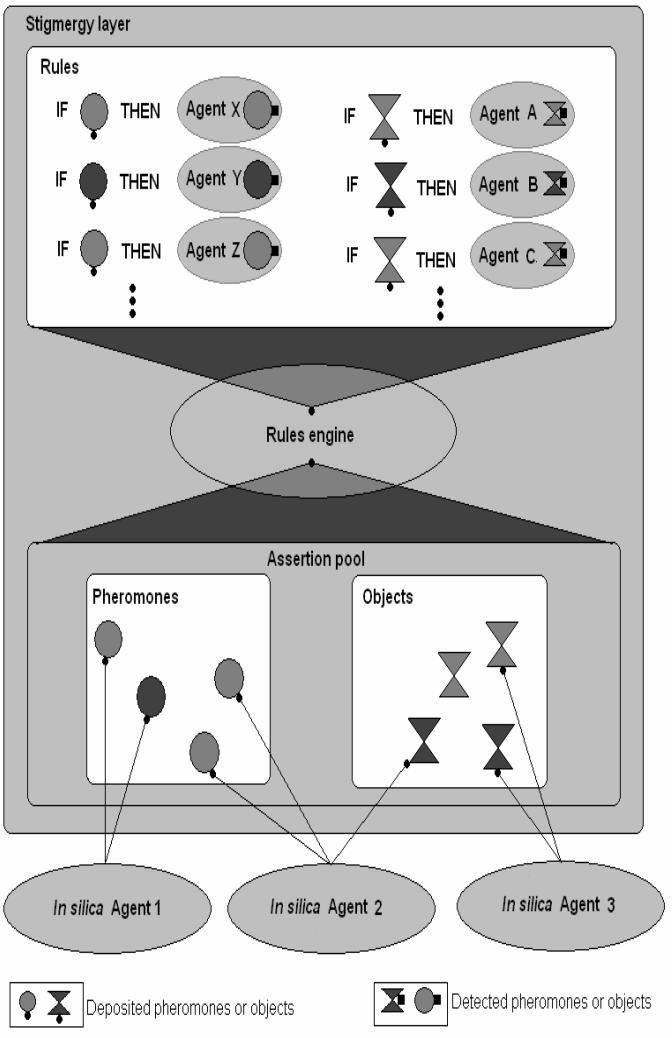

Fig. 3. Agents depositing pheromones and objects into the assertion pool of the rules engine. The assertion pool asserts these pheromones and objects into the rules engine that cause particular rules to fire. The rules then activate other agents. This mechanism is acting as if the activated agents actually detected the pheromones or objects.

The stigmergy layer in this framework acts as a single virtual agent. This virtual agent would be the environment

that all the agents are interacting with. This is analogous to the overall concept of an ant colony as the ants interact and communicate with the environment as if it were just another agent.

4.3 *In silica* agent layer

All the *in silica* agents in the framework are housed in the *in silica* layer. These agents are responsible for the key tasks in the environment being simulated. In a natural ant colony system these agents would represent the queen, drones, workers, soldiers and brooding ants. The activities of these agents ensure the survival and operation of the collective as a whole.

In silica agents communicate indirectly with each other via the stigmergy layer. Communication takes place by depositing pheromones or objects in the stigmergy layer. Communication between the *in silica* agents is vital in the ACEUS framework even though the agents are unaware of one another and the communication is indirect. In the ACEUS framework detecting different pheromones stimulates the agents. The response of the agent to the stimuli is governed by the rules built into the agent.

Every *in silica* agent in the framework inherits from a template agent. The template agent contains all functionality that is common among all the agents. The template agent will basically contain a rules engine utilising a RETE algorithm [15], a terminating mechanism and a communications mechanism. The rules engine will be used to assert and retract facts as well as fire the agent's rules. The rules engine acts as the agent's brain. The simple intelligent behavior displayed by the agent is a result of rules firing in the rules engine. The communication mechanism is utilised to deposit and detect pheromones or objects in the stigmergy layer. The communication mechanism utilises the rules engine in order to know how to respond to certain pheromones or objects in the stigmergy layer. Individual *in silica* agents will have their own set of rules that will be based on the concepts in their atom ontology. If a new agent is to be added to the framework, the designer simply has to inherit from the template agent and design an atom ontology and a new set of rules for that agent. An embedded rules engine is used instead of a global rules engine so that the agents can act as independent executing entities that may be distributed on different machines all over a computer network.

If a new agent is added to the system in order to extend the systems functionality it is simply added to the *in silica* layer. Thus no disassembling or redesign is needed when extending the system. The new agent only needs to know how to interact with the stigmergy layer and which responses it should have for specific pheromone stimuli. The ability to interact with the stigmergy layer will be encoded into the agent's rule set. This robustness and flexibility in the ACEUS framework is due the agents in the framework being unaware of each other. The only interaction or communication between the agents is via the stigmergy layer. If a commercial organization implementing the ACEUS framework needed to extend it's domain in order to incorporate new entities the framework would be adaptive and robust enough to handle the transition without any disassembling or redesign. Only the new agents with the knowledge of the new entities are added to the the *in silica* layer.

4.4 Rule base layer

The rules for the different agents are written into individual rule files. All the rule files in the ACEUS framework make up the rule base layer. When an agent is instantiated or activated the agent knows which rules file to execute in the rule base layer. The rules file can be edited i.e. new intelligence can be added to the agent dynamically without effecting other layers in the ACEUS framework.

The rule base layer also allows the designers easy access as well as dynamic access to the rule files. Dynamic access means that the designer can change the rules while the agent is executing. The new rules set or edited rule set will be executed when the agent refreshes or resets. A designer does not have to even stop or terminate the agent when editing the agent's rules set nor delve into the runtime code of each agent to change its intelligence as in present systems.

Rules form a powerful means for developing intelligence into different agents. The ACEUS framework allows rules to be defined that are general in the rule base layer. General rules are rules that are used by a number of agents in the framework i.e. this is shared intelligence. Designers only need import the file location of general rule set in order for the agent to utilse the general rules.

4.4 Ontology layer

The ontology layer constitudes the core layer of the ACEUS framework. Similar to the rule base layer the ontology layer is the ontology for the entire collective. The collective ontology of the ACEUS framework represents the ontology layer. Thus the ontology layer is an ensemble of atom ontologies. Atom ontologies are captured in the Web Ontology Language (OWL) [16] format in an XML files which are referred to as ontology files. The ontology layer is composed of all the ontology files in the ACEUS framework.

The ontology layer can also be described as the collective's vocabulary. Every entity and object known in the domain simulated by the ACEUS framework will be defined in the ontology layer. The ontology structure will help form the main blue print of the simulated system. The simulated system refers to the system domain being simulated by the ACEUS framework. In short the complete simulated system is modeled or characterised by means of the ontology.

What is interesting about the ontology layer and rule base layer is due to the distribution of the segments of both these layers into to the agents no agent will ever be able to comprehend the collective's intelligence or have an understanding of these layers. At most an individual agent will only be able to comprehend main chores and abilities which form a small segment of these layers. Thus the ACEUS framework forms a true analogy of an ant colony.

The ontology file for an agent is executed before the rules file when an agent is activated. This is due to the rules being dependent on the concepts and entities defined in the

ontology files. The ontology files contain concepts as well as services with specific URL locations. The concepts defined are entities that contain properties and are meant to simulate or mirror the real entities in the domain. The services are utilities that the agent can utilise to complete a specific task. The services are executable java applications that are encapsulated into a java archive files (jar files). These jar files are distributed over the computer network or web. The URL of the jar file is specified in the ontology. The agent can instantiate the classes for the service on distributed machines via HTTP. These services can be used by the agents to perform a magnitude of tasks e.g. connecting to databases, utilising TCP/IP, RMI, CORBA and communicating with vendor specific hardware.

5 CAS PROPERTIES AND MECHANISMS

In order to present ACEUS as a CAS, ACEUS needs to be described according to the properties and mechanisms of a CAS as defined by Holland [1]:

5.1 Aggregation (property)

The ontology layer in the ACEUS framework satisfies aggregation. According to Holland [1] aggregation has two instances. In the first instance familiar entities are categorized. In ACEUS all the entities in the domain to be simulated are defined and categorized in the ontology. The ontology layer is the definition layer of all entities in the domain. The ontology forms the vocabulary for the framework and all modeling of intelligence in the rules. The rules are highly dependant on the definition and categorization of the entities in the ontology layer. The second instance of aggregation is concerned with the emergent behavior that emerges from the aggregated interactions of the agents. The ACEUS framework has been designed so that there are unpredictable interactions both between agents and between agents and the environment. In ACEUS the individual agents have a very stereotype behavior and have no way of perceiving the complete collective. The aggregate in ACEUS framework would then be the collective agent layers, ontology layer and rule based layer that are highly robust and adaptive.

5.2 Tagging (mechanism)

The formation of aggregates and boundaries is consistently facilitated by the tagging mechanism [1]. Tags can be described as a mechanism of signaling between agents. This mechanism facilitates communication. Tags can also be seen as messages in different forms that agents can interpret allowing them to selectively interact with other agents or the environment. In ACEUS the utilization of deposited pheromones or objects in the stigmergy layer facilitates the signaling between the agents or between agents and the environment. The ACEUS framework thus utilises the tagging mechanism in the stigmergy layer.

5.3 Non-linearity (property)

The *in silica* agent layer in ACEUS is composed of many individual *in silica* agents. These *in silica* agents are event driven as they have embedded rules engine that fires particular rules when the necessary facts are asserted. This allows the *in silica* agents an undeterministic characteristic that is unpredictable. Similarly, the stigmergy layer in the ACEUS model also has an unpredictable nature. It too has a rules engine that facilitates event driven actions that are unpredictable. It can be concluded that the ACEUS framework has a non-linear property associated to due firstly to the actions of the *in silica* agents in the *in silica* agent layerbeing unpredictable or non-linear and secondly to the actions inside the stigmergy layer being unpredictable.

5.4 Flows (property)

Flows in CAS are referred to as the flow of an entity through a network of nodes and connectors. The nodes are the processors i.e. the agents in the model and the connectors are the designate possible interactions. Flows vary over time. Nodes and connectors can appear and disappear in a CAS over time. Thus neither the flows nor the network are fixed in time they are simply patterns that reflect a certain state of the model in time [1]. The tagging mechanism defines the network by creating major connections where the interactions are most critical.

ACEUS have objects and pheromones that flow through the system over time. No pheromone or object in the stigmergy layer is static in time. Each pheromone that is deposited in the stigmergy layer has a predefined lifetime as it begins to evaporate as soon as it is deposited. The objects do not evaporate but are consumed by the agents in the framework. The agent can consume the agent by retracting it from the stigmergy layer. The pheromones and objects in the stigmergy layer form a network of messages in the framework. However, the network is not static it constantly changes over time as the pheromones evaporate and the objects are consumed. As pheromones and signaling objects are used as tags in the ACEUS model they facilitate the creation of interactions between the nodes (agents) in the model.

5.5 Diversity (property)

The wide range of agents interacting in the framework satisfies the diversity property of ACEUS. ACEUS provides for organic agents, interfacing agents and *in silica* agents all of which are diverse in their own right. For example in the *in silica* layer of ACEUS there is a wide range of *in silica* agents, each filling a small niche that is defined by interactions depending on the agent. The rules embedded in the agent define the type of interactions the agent can engage in. Similarly in the organic layer there are a number of unique interfacing agents with completely different interactive roles.

The persistence of an individual agent depends on the ecosystem of agents that surround it. For example in the ACEUS framework if there is a void in the interaction of agents, an agent is created by the designer to fill in that void. The new agent may be a hybrid representation of other already existing agents i.e. the new agent might share some of the already existing agent's rule files and ontology files. This can be seen as convergence of the system. Convergence is the basis of emergent patterns that reappear again and again in widely disparate environments [1].

Mimicry can also be modeled with ACEUS by means of inheritance in the ontology layer. Agents can be mimicked in ACEUS by inheriting from another agent's atom ontology. This means rules that applied to the original agent will also apply to the mimicked agent. The rules are not changed only the atom ontology of the mimicked agent is different i.e. the mimicked agent's atom ontology might be a subset of its ancestor agent's atom ontology. The rules will fire as normal on the ontology segments.

5.6 Internal models (mechanism)

An ideal model of a CAS would be a system that could learn from experience and adapt its behavior. In order for a system to incorporate such behavior a basic mechanism is needed that ensures the ability to develop and act on internal models that simplify the external world.

The rules in the rule base of the ACEUS framework provide for this mechanism. The segmented rule base i.e. specific rules files that each agent executes gives the agent the ability to choose particular actions as a response to specific stimuli encountered. The rules allow the agents to infer the results of actions before they are taken allowing agents to select actions that would result in productive results.

5.7 Building blocks (mechanism)

A comprehensive feature of a CAS is the utilisation of building blocks to generate internal models [1]. Building blocks used to generate internal models could be the basis of dramatic improvements in software productivity [11]. Building blocks incorporated into a model ensure that the complexity is made more understandable and easier to manage.

The building blocks of the ACEUS framework are embedded in all the layers. The ontology layer is the layer in which all concepts of the environment are defined. The rule base layer is the layer where all the rules are built up from the concepts in the ontology layer. The *in silica* agents and interfacing agents contain the rules that are the building blocks of their intelligence. The aggregated intelligence of all the agents in the ACEUS framework builds up the collective's intelligence. Thus the importance of the building blocks mechanism in the ACEUS framework can be seen.

6 Conclusions

The ACEUS framework has been presented that can be used in a business environment to build a software system that imitates a complex adaptive system (CAS). The design of ACEUS entailed a number of layers. These layers allowed the system to be easily accessed from a designer's point of view. The layers allowed the framework to be easily changed without disassembling or redesigning any parts of the framework. The framework's design allowed easy adaptability in terms of adding new agents to the framework as well as an easy way to access the systems ontology and rule base without accessing the individual agent's internal code.

The ACEUS framework was created for environments that are growing at a rapid pace where the present methods of software design cannot keep up to the constant change. The present design structures in these environments are not robust or adaptable enough to cope with the constant change and expansion. The ACEUS framework allows reuse of code, agents, concepts and rules as well as dynamic addition of these entities into the framework. Designers can add entities to the framework without having to have a complete understanding of all the entities in the framework. The ACEUS framework is robust enough to be distributed over many machines or operate on a single machine. ACEUS allows thin client, thick client or web clients. For example interfacing agents can be designed to be simple HTML pages or full-blown applications.

One of the most promising ideas the ACEUS framework presents is that complex behavior patterns can emerge from individuals that are following simple rules. As the ACEUS framework grows i.e. more and more agents are added to the framework the complexity of the system can increase dramatically. To avoid this problem the complexity of the rule base and ontology is hidden from the individual agents. Individual agents can only view a small segment of the rule base and a small segment of the ontology. The segment the individual agent can comprehend is the segment that is actually executes. Thus a designer adding a new agent to the framework will only be concerned with the atom ontology and the rules of the agent. The agent designer does not need to be concerned with all the other ontologies and rules in the collective.

The ontology layer and rule base layer is abstract as a whole, as these two layers are an aggregate of many smaller segments. The complete ontology layer and rule base layer can only be viewed from the collective's perspective. By incorporating a collective's and an individual's perspective reduces complexity in the framework e.g. viewing the system from a collectives perspective could be overwhelming in complexity whereas the view point from the individuals perspective would be very simple and easy to understand.

The intention for future research is to implement ACEUS as a framework for the semantic web. Utilise ACEUS as a framework for building and maintaining peer-to-peer (P2P) networks. Build applications for monitoring and controlling complex environments such as cellular telecommunications network and modern pebble bed nuclear reactors. The main goals are to utilise the ACEUS framework to help build adaptive, robust and self-organizing systems that are able to handle the constant fluctuations and growth of modern business software.

Acknowledgment

The authors wish to thank the Rand Afrikaans University and the Department of Computer Science at the Rand Afrikaans University.

References

[1] J.H. Holland, *Hidden Order: How Adaption Builds Complexity*, Helix Books, 1995.

[2] J. Sutherland, "Business Object and Component Architectures: Enterprise Application Integration Encounters Complex Adaptive

Systems", HICSS-34 Outrigger Wailea Resort Maui, January, 2001, http://jeffsutherland.org/papers/HICSS2001/EAICAS.pdf, (URL accessed 2004, Jul).

[3] J. Sutherland and W. van den Heuvel, "Enterprise Application Integration Encounters Complex Adaptive Systems: A Business Object Perspective", 35th Annual Hawaii International Conference on System Sciences (HICSS'02) Big Island, Hawaii, Volume 9, January,2002, http://csdl.computer.org/comp/proceedings/hicss/2002/1435/09/14350286b.pdf, (URL accessed 2004, Jul).

[4] F. Heylighen, "Collective Intelligence and its Implementation on the Web: algorithms to develop a collective mental map", Computational and Mathematical Organizational Theory, 5(3): 53-280, 1999.

[5] L.K. Kristensen, "Aintz: Collective Problem Solving by Artificial Ants", http://www.evalife.dk/publications2.php, 2000, (URL accessed 2003, Aug)

[6] M. Dorigo and G. Di Caro, "The ant colony optimization meta-heuristic", *New Ideas in Optimization*, D. Corne, M. Dorigo and F. Glover, eds., McGraw-Hill, pp.11-32, 1999.

[7] M. Dorigo, V. Maniezzo and A. Colorni, "Ant system: Optimization by a colony of cooperating agents", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 26, no. 1, pp.29–41, Feb 1996.

[8] M. Dorigo and L. M. Gambardella, "Ant colonies for the traveling salesman problem", BioSystems, vol. 43, pp.73–89, 1997.

[9] R. Schoonderwoerd, O. Holland, J. Bruten and L. Rothkrantz, "Ant-based load balancing in telecommunications networks", Adaptive Behavior, HP Labs Technical Report, HPL-96-76, May, 1996.

[10] E. Bonabeau, F. Henaux, S. Guerin, D. Snyers, P. Kuntz and G. Theraulaz, "Routing in telecommunications networks with smart ant-like agents", *Proc. of Intelligent Agents for Telecommunications Applications*, 1998.

[11] E. Bonabeau and C. Meyer, "Swarm Intelligence: A Whole New Way to Think About Business", Harvard Business Review, Harvard Business School Publishing Corporation, 2001.

[12] P. Valckenaers, M. Kollingbaum, H. Van Brussel, O. Bochmann and C. Zamfirescu, "The Design of Multi-Agent Coordination and Control Systems using Stigmergy", Proc. of the IWES'01 Conference,2001,http://www.csd.abdn.ac.uk/~mkolling/publications/DesignMultiAgentCoordControlStigmergy.pdf, (URL accessed 2004, Jul).

[13] J.H. Holland, *Emergence: from Chaos to Order*, Oxford University Press, 1998.

[14] S. Johnson, *Emergence: The Connected Lives of Ants, Brains, Cities and Software*, Scribner, 2002.

[15] C.L. Forgy, "Rete: A Fast Algorithm for the Many pattern/Many Object-Pattern Matching Problem", Artificial Intelligence, Vol. 19, no. 1, September, pp 17-37, 1982.

[16] M.Dean and G.Schreiber, "OWL Web Ontology Language Reference", W3C Working Draft, http://www.w3.org/TR/owl-ref/, 2003, (URL accessed 2004,Jun).

G.B. O'Reilly is a PhD student in Computer Science at the Rand Afrikaans University (RAU). His research interests include swarm intelligence, complex adaptive systems and application of these technologies to new fields of research such as the semantic web. He received his MSc degree in physics in 1994 and his MSc in computer science in 2003 both from RAU. He works as a software engineer at Mobile Telecomunications Network (MTN) in South Africa.

E.M. Ehlers is a professor of computer science at the at the Rand Afrikaans University (RAU). Her research interests include agent technology. She holds a PHd. in computer science received in 1987 from RAU.