Chapter 1

Introduction

1.1 Motivation

Today's software applications increasingly rely on wireless devices – such as, PDAs, laptops, mobile phones or sensors – interacting each other on top of novel infrastructures (e.g. sensor networks, ad-hoc networks). As time goes by, the number of these devices in daily life increases continuously providing powerful infrastructures that are not widely exploited due to current software limitations (scalability, realtime requirements, or failure tolerance). Moreover, devices are becoming smaller, making possible new infrastructures such as smart sensor networks, smart materials, self-reconfiguring robots, collaborative unmanned aerial vehicles, or self-assembling nanostructures that were science fiction until now. These new structures allow the implementation of a wide range of new application and services, such as, rescue applications (search of survivor or ad-hoc communication infrastructures), disaster prevention (toxic clouds monitorisation, Earthquake and tsunami detection, or forest fires monitorisation) or act in favor of the environment (pollution sources detection, acoustic sources detection, oil's leaks on the oceans or traffic light control).

Such infrastructures are characterised by a great deal of *openness* (the number of entities in the system can change), *large scale* (thousands of nodes), *dynamism* (the network topologies are changing along the time), and *unpredictability* (environment changes, failures, or new entities joining), which cannot be coped with traditional or centralized approaches to system design or engineering.

To deal with these features, a new software tendency is to provide entities in the system with autonomy and pro-activity and to increment the interaction between them. This betting on incrementing interaction and decentralising responsibilities over those entities provides systems with better scalability, robustness, and reduces the computation requirements of each entity. Moreover, by increasing the interaction the system reduces the computation power required by each entity. Moreover, it also permits to reduce, even more, the devices'

sizes making possible new structures commented above. According to this new paradigm, so-called *Self-Organising Multi-Agent Systems*, the desired goal or behaviour emerges from the local interactions rather than from a centralised entity. Moreover, the intelligence resides in the interactions among the agents and not in each individual model (i.e. intelligence is collective). Thus, in spite of individual agents being simple (i.e. while they work individually), the agents can overtake the individual limitations and achieve complex tasks in a collaborative manner.

Self-Organisation in computer science is described as follows:

"A system described as self-organizing is one in which elements interact in order to achieve dynamically a global function or behavior." (Carlos Gershenson 2007)

Biological systems have been adopted as a source of inspiration for Self-Organising systems. Biological self-organising systems are present in the nature in a wide range of pattern formation processes, such as, fish swimming in coordinated schools, flocking behaviour in birds, patterns on seashells, synchronous fireflies flashing, etc.... In all of them the desired behaviour emerges from the coordination between the individuals using only local interactions, and local knowledge. A more general definition of Self-organising system is:

"Self-organization systems are physical and biological systems in which pattern and structure at the global level arise solely from interactions among the lower-level components of the system. The rules specifying interactions among the systems's components are executed using only local information, without reference to the global pattern." (Scott Camazine 2006)

Since long, Self-Organisation has been mainly discussed in physics and biology. Nowadays, self-organising systems are applied to Multi-Agent Systems (MAS) to achieve robustness, failure tolerance, scalability and adaptability. A variety of self-organising, bio-inspired mechanisms have been applied in different domains, achieving results that go beyond traditional approaches [Mamei et al., 2006]. However, researchers usually apply these mechanisms in an ad-hoc manner. In this way, their interpretation, definition, boundary, and implementation typically vary among the existing literature preventing these mechanisms from being applied clearly and systematically to solve recurrent problems.

Two main challenges appear when Self-Organising mechanisms are applied in MAS: (1) to control the emergent behaviour of the system from the local interactions and local knowledge of the environment avoiding uncontrolled emergent behaviours, and (2) engineering self-organising MAS in a systematic way. These

challenges are addressed by self-organising systems engineering.

The idea of engineering self-organising systems in computer science has attracted different researchers recently. Nagpal et al. [Nagpal, 2004] presented a set of biologically-inspired primitives that describe how organising principles from multi-cellular organisms may apply to large scale multi-agent systems. Those primitives have emerged as part of the Amorphous Computing project [Abelson et al., 2000a], which is focused on developing programming methodologies for systems composed of vast numbers of locally-interacting and identically programmed agents. That work was motivated by new emerging technologies such as MEMS (micro-electronic mechanical devices), which enable to create tiny computing and sensing elements that can be embedded into materials, structures or the environment. Some envisioned applications are: reconfigurable robots or structures composed of millions of identical modules that self-assemble into different shapes to achieve different tasks, smart environments where the sensors are embedded into the walls, or armies of ants-like robots that can achieve complex tasks in a collaborative way. That work was a first attempt towards assembling a catalog of primitives for large scale multi-agent control, where the desired behaviour emerges from the local interaction and coordination between agents. Moreover, the primitives are interesting from the application point of view and it was a step forward for engineering self-organising systems. However, those primitives are not presented together with an implementation process or by taking into consideration the different scenarios where the primitives can be applied. Thus, it is difficult to use them in a systematic way.

Mamei et al. [Mamei et al., 2006] presented a review on the state of the art of nature-inspired self-organising mechanisms in computer science and proposes a taxonomy to classify those self-organising mechanisms found in the literature. As previous works, the mechanisms are described together with the biological process they were inspired from. Moreover, they add the different domains that can be addressed with each mechanism. That work was the most complete catalog at that time, and certainly it was a step forward for engineering bio-inspired self-organising systems. Even when these descriptions can drive the implementation of the mechanisms, they are far away from being considered as set of mechanisms that can be applied in a systematic manner. However, that work motivates to go further and propose new questions:

- Which are the problems that those mechanisms can solve?
- What solution contributes each pattern?
- What are the main trade-offs to consider in the implementation?

To answer those questions and make the self-organising mechanisms applicable more systematically, different authors have focused on proposing descriptions of self-organising mechanisms under the form of software design patterns. Software design patterns was proposed by [Gamma et al., 1995], [Buschmann et al., 1996] and [Lind, 2003] for object oriented software. In software engineering, a pattern design describes a reusable solution for a

commonly occurring problem. Focusing on self-organising mechanisms, the idea of the design pattern structure makes it easy to identify the *problems* that each mechanism can solve, the specific *solution* that it brings, the *dynamics* among the entities and the *implementation*. Thus, self-organising systems can be designed more systematically.

In [Babaoglu et al., 2006], based on the existing software design patterns, they proposed a conceptual framework for transferring knowledge from biology to distributed computing. The mechanisms proposed in that paper are described using some problems that are solved with each pattern, the solution that each pattern provides and the biological process where it was inspired. Another relevant idea presented in [Babaoglu et al., 2006] paper is the distinction between basic patterns and composite patterns. However, they presented only one composite pattern (chemotaxis) and it was not really presented as a composition, it was an extension or application using one basic pattern.

Following the idea proposed by [Babaoglu et al., 2006], i.e. some patterns are composed by basic ones. Gardelli et al. [Gardelli et al., 2007] propose a set of basic design patterns for Self-Organising Multi-Agent Systems that can be combined to create a well known mechanism. The idea related with the basic patterns is to get a deeper understanding of the systems dynamics and also improve the controlability. The basic patterns properly combined produce new complex patterns and make easy to adapt the existing patterns to a new problems. As far as we know, [Gardelli et al., 2007] was the first to decompose a complex pattern into basic ones. Specifically, the Stigmergy Pattern was decomposed in evaporation, aggregation, and diffusion patterns. The stigmergy pattern is a coordination mechanism, based on indirect communication where the agents are mobile and communicate each other by modifing variables that are located in the environment. The name of stigmergy comes from the greek stigma (mark, sign) ergon (work, action) and contains the idea that the agents leave the marks in the environment that stimulate the behaviour of other agents. The stigmergy was first time used to describe the ants behaviour and is in the Ants Colony Optimisation (ACO) algorithm where the stigmergy has been more fruitful.

The decomposition process consists on identifying the internal mechanisms existing in a complex mechanism and on observing the contribution that these internal mechanisms provide to the complex one. In this example the stigmergy pattern is decomposed in evaporation, diffusion, and aggregation patterns. In [Gardelli et al., 2007], they keep the idea to use design patterns as conceptual framework to describe self-organising mechanisms.

The patterns proposed in [Gardelli et al., 2007] paper are all related with the ant colonies behaviour. The model provided presents too many constraints to be generalised and the examples of usage are not related to engineered self-organising systems. Thus, it is still an open question if basic patterns can be used isolated in self-organising systems and if basic patterns can be used to compose other self-organising patterns different than the pattern where they come from.

Another interesting work where a set of mechanisms are presented as design patterns is presented in [De Wolf and Holvoet, 2007], where they discussed an extended catalogue of mechanisms as design patterns for self-organising emergent applications. The patterns are presented in detail and can be used to systematically apply them for engineering self-organising systems. However, relations among the patterns are missed, i.e. the authors do not describe how patterns can be combined to create new patterns or adapted to tackle different problems. Moreover, in the paper one of the mechanisms proposed is applied to a case study, "A packet delivery service", but not compared with existing approaches.

Based on the set of mechanisms proposed in [Mamei et al., 2006], Sudeikat et al. [Sudeikat and Renz, 2008] discuss how intended MAS dynamics can be modeled and refined to decentralised MAS designs, proposing a systematic design procedure that is exemplified in a case study. Until now the only research focus to decompose complex patterns in basic ones was done by [Gardelli et al., 2007] as we commented before. The decomposition of complex patterns has not widely exploited and it is still an open issue. Thus, some of the questions that this book tries to answer are:

1. Can other complex patterns be decomposed?
2. Can the basic patterns be used isolated and make contributions in known problems?
3. Can the basic patterns be used to compose several complex patterns?

None of those papers have implemented the patterns to solve existing problems and have demonstrated the contribution of the patterns in different existing problems. Although, Self-Organising mechanisms described as design patterns provide useful descriptions that help to clarify the definitions of these mechanisms. These efforts are still fragmented: no clear catalogue of these patterns is provided, interpretations vary among authors, or the relations among patterns and their precise boundaries are not described. Moreover, they are far away for being applied systematically.

1.2 Contributions

In this book, we focus on modeling bio-inspired mechanisms for engineering self-organising systems by design patterns, arguing that some mechanisms can be described in terms of basic ones, i.e. fundamental mechanisms that can be used alone or as a part of more complex patterns. This decomposition of the self-organising mechanisms permits the identification of their exact boundaries, the relations they have with basic mechanisms, and to use them to compose new mechanisms or adapt them to acquire the desired behaviour. Every pattern is provided with a detailed description of the problem that it focus on, the corresponding solution that each pattern provides, and their behaviors (interaction dynamics and algorithmic behavior). Unlike the existing works, the basic patterns proposed in this book have demonstrated to be able to compose more than

one pattern allowing, in this way, to easily create new composed patterns in a systematic way. On the other hand, this book presents the generality of some of these patterns by applying them to different areas such as, dynamic optimisation, spatial computing, or sensor networks. The use of these basic patterns has improved the performance in the different areas compared with the existing techniques, demonstrating their general purpose.

Structuring mechanisms as design patterns allows a better support to create new mechanisms and to adapt existing ones to solve new problems. Moreover, this structure also allows a clear identification and separation of the mechanisms appropriate to each pattern.

To evaluate the performance of the proposed patterns, this book focuses on three different domains where the patterns are applied: (1) Dynamic and Noisy Optimisation, (2) Spatial Computing, and (3) Sensor Networks. The goal of this book is to propose general-purpose self-organising patterns and to show how these patterns can be applied to existing problems making contributions to different fields.

The main contributions of this book are the following:

- A set of complex bio-inspired self-organising mechanisms are presented as design patterns for engineering self-organising systems. These patterns are generalised and classified from their existing applications in the literature.
- Complex bio-inspired design patterns are decomposed in basic patterns. Basic patterns are presented also as design patterns that can be combined to create new patterns or to adapt the existing patterns to resolve new problems.
- We propose a model where bio-inspired patterns can be defined. The model covers a wide set of applications found in the literature.
- Some of the patterns proposed in this book have been implemented, making contribution in different communities. The communities and their contributions based on the pattern are:
  - Dynamic Optimisation: Adaptation of Particle Swarm Optimisation for working in noisy and dynamic environments. The existing Particle Swarm Optimisation algorithm has been extended with the Evaporation Pattern improving its performance in dynamic and noisy optimisation.
  - Spatial Computing: New set of algorithms for infrastructureless spatial storage of information. These new algorithms exploit mobile devices located in the environment. In these algorithms the agents decide when to replicate and collaborate between them to ensure the information coverage in a specified area.
  - Sensor Networks: Multi Mobile Agent approach to locate diffuse event sources using a Wireless Sensor Network infrastructure. Locating and tracking of diffuse event sources problem is proposed as a new

interesting problem with important application in real word domains. The mechanism used to tackle this new problem has demonstrated to achieve a good performance and presents good tolerance in front of sensor network failures and noisy and dynamic environments

1.3 Book structure

This book is organised as follows:

- Chapter 2: Bio-inspired design patterns.

  The goal of this chapter is to provide a complete catalog of bio-inspired mechanisms existing in the literature, their relation, their boundaries and the problem that each mechanism is focused on. Specifically, we analyze bio-inspired self-organising mechanisms existing in the literature and present these mechanisms as design patterns that can be applied in a systematic manner to engineer self-organising systems. The patterns are classified and the relations between them are identified. To describe the dynamics and interactions between the entities participating in each pattern, a computational model is proposed.

- Chapter 3: Dynamic Optimisation.

  In this chapter the Evaporation Pattern, proposed in Section 2.3.4, is incorporated to the Particle Swarm Optimisation (PSO) algorithm to deal with dynamic and noisy optimisation problems. The new algorithm proposed is compared to the existing version without evaporation. Benchmark results are provided, demonstrating that the evaporation mechanism allows the PSO algorithm to improve its performance in dynamic optimisation problems, mainly, when the fitness function is subjet to noise.

- Chapter 4: Hovering Information in Spatial Computing.

  In this chapter we define and analyse a collection of algorithms based on the Replication Pattern (Section 2.3.2) and the Repulsion Pattern (Section 2.3.5), for persistent storage of information at specific geographical zones exploiting the resources of mobile devices located in these areas. This proposed application is an example of problem difficult to trackle with traditional approaches due to the large number of nodes and real time requirements. Performed experiments and study of algorithms' parameter are analysed.

- Chapter 5: Detecting Dynamically Changing Diffuse Event Sources in Noisy WSN Environments.

  This chapter proposes and evaluates the use of the Chemotaxis Pattern applied in sensor networks for localizing dynamically changing diffuse events. Aimed in reducing the power consumption of the sensors, mobile agents collaborate to find diffuse event sources in dynamic and noise environments.

The goal is to locate the diffuse event sources using a minimum number of messages and sensors' reads. Conducted experiments demonstrate that the proposed approach is able to find the sources even in presence of noise, using an aceptable number of messages and sensors' read.

- Chapter 6: Conclusions and Future Work. Main conclusions and open research lines are detailed in this Chapter.