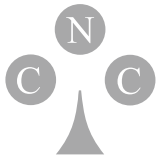
Natural Computing Series



Peter R. Lewis • Marco Platzner • Bernhard Rinner
Jim Tørresen • Xin Yao
Editors

Self-aware Computing Systems

An Engineering Approach

Springer

Editors
Peter R. Lewis
School of Engineering & Applied Science
Aston University
Birmingham, United Kingdom

Bernhard Rinner
Institute of Networked and Embedded Systems
Alpen-Adria-Universität Klagenfurt
Klagenfurt am Wörthersee
Austria

Xin Yao
School of Computer Science
University of Birmingham
Birmingham, United Kingdom

Marco Platzner
Department of Computer Science
Paderborn University
Paderborn, North Rhine-Westphalia
Germany

Jim Tørresen
Department of Informatics
University of Oslo
Oslo, Norway

ISSN 1619-7127
Natural Computing Series
ISBN 978-3-319-39674-3 ISBN 978-3-319-39675-0 (eBook)
DOI 10.1007/978-3-319-39675-0

Library of Congress Control Number: 2016942574

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG Switzerland

Foreword

This book considers the design of new computation systems that are in some ways more responsive to the environment and their own state than current system designs and aim to be more reliable through the creation of self-aware and self-expressive systems. One of the driving forces of this work is the realisation of the growth in system complexity and the difficulty of using current "standard" methods and designs to continue to create working systems. This is certainly relevant, as the interest in the design and understanding of complex computing systems in technical applications has been growing significantly in various research initiatives like autonomic, organic, pervasive or ubiquitous computing and in the multi-agent system community. There have been many novel applications demonstrating a wide range of self-* properties, as well as studies looking also at emerging global behaviour due to self-organised local interaction. The authors of this book present the results of a large European cooperative project focusing specifically on self-awareness, which may be seen as one of the essential backgrounds for developing and supporting the other self-* properties, which is addressed here by the term "self-expression".

Ever since researchers have realised that machines could be programmed to have increasingly adaptive behaviors, there has been much research on how to introduce adaptive behaviour and more biological like capabilities into systems – more types of reasoning, more types of awareness, and more types of intelligent processing. Particularly important in adaptation is that the system has the knowledge and the capabilities that allow it to do these adaptations in novel situations and at runtime. There are many examples of large-scale programmes to foster the understanding of the necessary attributes and architectures of systems capable of these adaptations. Hence there were programs on adapting routers and networks in real time (e.g., DARPA's Active Networks), platforms and other plug and play architectures with robust real time services (e.g., DARPA's META program; Europe's AUTOSAR (AUTomotive Open System ARchitecture), programs that worked to understand emergent behaviour and make use of it (Europe's Organic Computing), systems with computational reflection used for resource management (e.g., reflective architectures), and, of course, an enormous amount of work on multi-agent systems and autonomous computing.

In this landmark EU project, these slowly developing themes, drawn from a wide diversity of fields, have been brought together and further developed with both thoughtful discussions on foundations and new research and developments in the engineering of several application areas.

One particularly important aspect of this book is the way in which it builds up our repertoire of engineering methods for self-awareness by purposely drawing its concepts for self-awareness from a diversity of fields and its examples from a diversity of applications. Most importantly, these applications span across different levels of computational systems, from agents and applications (interactive music systems in Chapter 14) to middleware services (Chapter 11) to adaptive networks (Chapter 10) and even hardware (Chapters 8, 9 and 12.)

Starting from insights into "self-awareness" achieved by other disciplines like psychology and philosophy, the notions of "computational self-awareness" and "self-expression" are systematically developed. The majority of the book focuses on computational systems that require some form of anticipation where the new algorithms and methods are needed to provide the appropriate anticipatory behaviour. In practice, these methods can include different forms of self-awareness (such as awareness of goals, of the current state and readiness of system resources, of one's planning process and of the ordering of events), such that the system is not simply reacting to events and changes, but can anticipate them. The ideas and mechanisms outlined are applied to a number of interesting applications: Computational finance applications using heterogeneous computing clusters are investigated and include self-adaptive algorithms that are supported by hardware; low-latency adaptive network processing; run-time reconfigurable hardware acceleration; heterogeneous computing and hardware/software co-processing for algorithmic trading and reconfigurable hardware acceleration of self-optimisation of reconfigurable hardware designs. Self-awareness in distributed smart camera networks is considered for both single cameras at a node level and multiple camera systems within a network. Interesting bio-inspired methods are aimed at the network level, artificial pheromones are employed to construct a local neighbourhood graph, allowing adaptation in the network as topologies change. A hypermusic demonstrator is considered as a third application. This considers various methods and techniques to enable adaptability (self-expression) in musical output. Three methods within this application are the focus of this work, each providing different input information and overall levels of information: *SoloJam* provides rather overarching rhythmic shaping; *Funky Sole Music* provides what might be considered more specific, lower level, inputs such as walking tempo, movement types and foot activity; *PheroMusic* considers more links between musical soundscapes.

Thus, the book provides a comprehensive introduction to self-aware computation providing a broad range of new theoretical background and foundation before moving on to consider details of architectures and techniques to help design self-aware computational systems, from nodes to networks. Many of the problems that have been addressed in this book will continue to be timely for many years to come and could well provide the focus of research strands within many research fields. Particular challenges remain with respect to performance, safety and security properties

of such systems. Although self-awareness is supposed to improve the performance of computational systems in complex environments, there is still a lack of formal frameworks for rigorously arguing about the behaviour of such systems.

The authors are all well known in this research area and the editors, Lewis, Platzner, Rinner, Torresen and Yao have done an excellent job in pulling together what is an excellent book.

Los Angeles — *Kirstie Bellman*
Karlsruhe — *Hartmut Schmeck*
York — *Andy Tyrrell*

March 2016

Preface

Self-aware computing is an emerging field of research. It considers systems and applications able to proactively gather and maintain knowledge about aspects of themselves, learning and reasoning on an ongoing basis, and finally expressing themselves in dynamic ways, in order to meet their goals under changing conditions. The aspects they might be aware of include their own internal state, capabilities, goals, environment, behaviour and interactions. The presence of gathered knowledge permits advanced intelligent decision making leading to self-expression: that is, effective, autonomous and adaptive behaviour, based on self-awareness. Self-awareness and self-expression capabilities are key to designing and operating future computing systems that will inherently and autonomously deal with high levels of dynamics and uncertainty, heterogeneity, scalability, resource constraints and decentralisation. Concepts of self-awareness have been established in psychology, philosophy and cognitive science but are relatively new to computing. In computing systems, our concepts of self-awareness and self-expression integrate and enhance a number of recent approaches dealing with systems with so-called self-* properties, e.g., self-adaptation, self-organisation and self-healing.

This book is the first ever to focus on the emerging field of self-aware computing from an engineering perspective. It first comprehensively introduces fundamentals for self-awareness and self-expression in computing systems, proposing the new notion of *computational self-awareness*. It then focuses on architectures and techniques for designing self-aware computing systems at the node and network levels. Finally, the effectiveness of these techniques is demonstrated on a variety of case studies. While a number of books on related topics such as self-adaption and self-organisation, and even self-awareness concepts in computing, have already been published, this book is unique as it provides a holistic view of self-aware computing including its relationship with self-expression, and the process of engineering such systems, i.e., a thorough understanding of how to model and build self-aware computing systems based on design patterns and techniques.

This book targets graduate students and professionals in the fields of computer science, computer engineering, and electrical engineering, but also practitioners and scientists from other fields interested in engineering systems with advanced proper-

ties relying on their ability to reason about themselves in a complex environment. The authors and editors of this book are active researchers in various aspects related to self-aware computing systems. They have a strong track record in successfully collaborating on this topic, for example, through the European FET project "Engineering Proprioception in Computing Systems (EPiCS)". The extensive joint experience of the contributors makes this edited book consistent and well integrated. Therefore, we specifically recommend this book as reading material for the graduate level or for self-study on self-aware computing systems.

The book reports some of the latest results in self-aware and self-expressive computing, and we hope it serves as a launchpad for further research discussions and new ideas in the future.

Birmingham — *Peter R. Lewis*
Paderborn — *Marco Platzner*
Klagenfurt — *Bernhard Rinner*
Oslo — *Jim Tørresen*
Birmingham — *Xin Yao*

March 2016

Acknowledgements

The research leading to many results in this book was conducted during the EPiCS project (Engineering Proprioception in Computing Systems) and received funding from the European Union Seventh Framework Programme under grant agreement no. 257906.

The contributors would like to acknowledge additional support for research performed in individual chapters of this book.

- Chapters 6 and 7 were also supported by EPSRC Grants (Nos. EP/I010297/1, EP/K001523/1 and EP/J017515/1).
- Chapter 8 was also supported by the German Research Foundation (DFG) within the Collaborative Research Centre "On-The-Fly Computing" (SFB 901) and the International Graduate School on Dynamic Intelligent Systems of Paderborn University.
- Chapter 9 was also supported in part by HiPEAC NoE, by the European Union Seventh Framework Programme under grant agreement numbers 287804 and 318521, by the UK EPSRC, by the Maxeler University Programme, and by Xilinx.
- Chapter 12 was also supported in part by the China Scholarship Council, by the European Union Seventh Framework Programme under grant agreement numbers 287804 and 318521, by the UK EPSRC, by the Maxeler University Programme, and by Xilinx.
- Chapter 13 was also supported by the research initiative Mobile Vision with funding from the Austrian Institute of Technology and the Austrian Federal Ministry of Science, Research and Economy HRSMV programme BGBl. II no. 292/2012.
- Chapter 14 was also supported by the Research Council of Norway under grant agreement number 240862/F20.
- Peter Lewis would like to thank the participants of the Dagstuhl Seminar "Model-Driven Algorithms and Architectures for Self-aware Computing Systems", Seminar Number 15041, for many insightful discussions on notions of self-aware computing.

Contents

List of Contributors

Andreas Agne
Paderborn University, Germany. e-mail: agne@upb.de

Rami Bahsoon
University of Birmingham, UK. e-mail: r.bahsoon@cs.bham.ac.uk

Tobias Becker
Imperial College London, UK. e-mail: tobias.becker@imperial.ac.uk

Arjun Chandra
Studix, Norway. e-mail: arjun@studix.com

Renzhi Chen
University of Birmingham, UK. e-mail: rxc332@cs.bham.ac.uk

Tao Chen
University of Birmingham, UK. e-mail: t.chen@cs.bham.ac.uk

Stewart Denholm
Imperial College London, UK. e-mail: stewart.denholm10@imperial.ac.uk

Bernhard Dieber
Alpen-Adria-Universität Klagenfurt, Austria. e-mail: bernhard.dieber@counity.at

Lukas Esterle
Alpen-Adria-Universität Klagenfurt, Austria. e-mail: lukas.esterle@aau.at

Gustavo Fernández Domínguez
Austrian Institute of Technology, Austria. e-mail: gustavo.fernandez@ait.ac.at

Funmilade Faniyi
University of Birmingham, UK. e-mail: f.faniyi@gmail.com

Andreea-Ingrid Funie
Imperial College London, UK. e-mail: andreea.funie09@imperial.ac.uk

Kyrre Glette
University of Oslo, Norway. e-mail: kyrrehg@ifi.uio.no

Ce Guo
Imperial College London, UK. e-mail: ce.guo10@imperial.ac.uk

Markus Happe
ETH Zurich, Switzerland. e-mail: markus.happe@alumni.ethz.ch

Maciej Kurek
Imperial College London, UK. e-mail: mk306@imperial.ac.uk

Peter R. Lewis
Aston University, UK. e-mail: p.lewis@aston.ac.uk

Achim Loesch
Paderborn University, Germany, e-mail: achim.loesch@upb.de

Wayne Luk
Imperial College London, UK. e-mail: w.luk@imperial.ac.uk

Leandro L. Minku
University of Leicester, UK. e-mail: leandro.minku@leicester.ac.uk

Georg Nebehay
Austrian Institute of Technology, Austria. e-mail: gnebehay@gmail.com

Xinyu Niu
Imperial College London, UK. e-mail: niu.xinyu10@imperial.ac.uk

Kristian Nymoen
University of Oslo, Norway. e-mail: kristian.nymoen@imv.uio.no

Roman Pflugfelder
Austrian Institute of Technology, Austria. e-mail: roman.pflugfelder@ait.ac.at

Marco Platzner
Paderborn University, Germany. e-mail: platzner@upb.de

Christian Plessl
Paderborn University, Germany. e-mail: christian.plessl@uni-paderborn.de

Bernhard Rinner
Alpen-Adria-Universität Klagenfurt, Austria. e-mail: bernhard.rinner@aau.at

Mark Salmon
University of Cambridge, UK. e-mail: mhs39@cam.ac.uk

Jennifer Simonjan
Alpen-Adria-Universität Klagenfurt, Austria. e-mail: jennifer.simonjan@aau.at

Stephan C. Stilkerich
Airbus Group Innovation, Germany. e-mail: stephan.stilkerich@airbus.com

Tim Todman
Imperial College London, UK. e-mail: timothy.todman@imperial.ac.uk

Jim Tørresen
University of Oslo, Norway. e-mail: jimtoer@ifi.uio.no

Ariane Trammell-Keller
ETH Zurich, Switzerland. e-mail: ariane.trammell@alumni.ethz.ch

Shuo Wang
University of Birmingham, UK. e-mail: s.wang@cs.bham.ac.uk

Xin Yao
University of Birmingham, UK. e-mail: x.yao@cs.bham.ac.uk

Acronyms

ACO Ant Colony Optimisation
AES Advanced Encryption Standard
ALA Ant Learning Algorithm
API Application Programming Interface
BSD Berkeley Software Distribution
CDC Concept Drift Committee
CDT Correct Detected Track
CMT Consensus-Based Matching and Tracking
CPU Central Processing Unit
CUDA Compute Unified Device Architecture
CV Computer Vision
DDD Diversity for Dealing with Drifts
DDM Drift Detection Method
DPS Dynamic Protocol Stack
DWM Dynamic Weight Majority
EA Evolutionary Algorithm
EDDM Early Drift Detection Method
EGO Efficient Global Optimisation
FAT False Alarm Track
FB Functional Block
FF Flip-Flop
FPGA Field-Programmable Gate Array
FPS Frames per Second
FMC FPGA Mezzanine Card
FOV Field of View
FSR Force-Sensitive Resistor
GA Genetic Algorithm
GPSP General Purpose Sensor Platform
GP Gaussian Process
GPU Graphics Processing Unit
H2S Hardware-to-Software

HLS High Level Synthesis
HMM Hidden Markov Models
HPC High Performance Computing
ICAP Internal Configuration Access Port
IDP Information Dispatch Point
ILP Integer Linear Programming
IP Internet Protocol
IPC Inter-process Communication
LUT Look-up Table
MAC Media Access Protocol
MLO Machine Learning Optimiser
MOEA/D Multi-objective Evolutionary Algorithm Based on Decomposition
MOP Multi-objective Optimisation Problem
MPI Message Passing Interface
MTBF Mean Time Between Failures
NoC Network-on-Chip
OSC Open Sound Control
OT Object Tracking
PE Processing Element
RAP Redundancy Allocation Problem
RAM Random Access Memory
RTM Reverse Time Migration
S2H Software-to-Hardware
SA Self-aware
SACS Self-aware Computing Systems
SDRAM Synchronous Dynamic Random Access Memory
SE Self-expression
SIMD Single Instruction, Multiple Data
SMT Satisfiability Modulo Theories
SVM Support Vector Machine
SoC System-on-Chip
SOP Single-Objective Optimisation Problem
SSE Streaming SIMD Extensions
STEPD Statistical Test of Equal Proportions
TCP Transmission Control Protocol
TDF Track Detection Failure
TPOT-RL Team-Partitioned Opaque-Transition Reinforcement Learning
Todi Two Online Classifiers for Learning and Detecting Concept Drift
TLD Tracking-Learning-Detection
UDP User Datagram Protocol
VHDL Very High Speed Integrated Circuit Hardware Description Language

Glossary

This glossary lists important terms used in this book, in particular in Part I "Concepts and Fundamentals", with accompanying descriptions or definitions. The glossary is organised into four sections: concepts of self-awareness and self-expression, engineering self-aware systems, related approaches, and general terms. The terms in each of the sections are listed alphabetically.

Concepts of Self-awareness and Self-expression

self-awareness Self-awareness is a broad concept which describes the property of a system (typically a human) which has knowledge of "itself", based on its own senses (perceptual) and internal models (conceptual). This knowledge may take different forms (cf. levels of self-awareness), and be based on perceptions of both internal and external phenomena (cf. public vs. private self-awareness). It can be a property of single systems (e.g., agents) and collective systems.

collective self-awareness Collective self-awareness refers to the self-awareness property of a collective system, i.e., as opposed to a single agent. Levels of, and public/private self-awareness apply also at this abstraction. This means that a self-aware system is not required to have a central "knowledge" component (though it may have, if desired).

computational self-awareness Computational self-awareness is a notion we have developed to refer to a computational interpretation of self-awareness. Since much of the literature on self-awareness does not readily make sense to engineers or applies directly to technical systems, aspects of computational self-awareness are designed to describe self-awareness properties of computational systems, inspired by self-awareness in humans.

emergent self-awareness This is a special case of collective self-awareness, when the collective self-awareness properties are present, but it is not obvious how this comes about by simply examining the behaviour of individual nodes within a collective.

level(s) of self-awareness A very common theme in self-awareness theory is the distinction between several levels of self-awareness, to describe different aspects or capabilities which comprise a system's complex self-awareness. There are many examples of "sets of levels" to be found in the literature. In developing our notion of computational self-awareness, we have based a set of levels of computational self-awareness on the set of levels (for humans) proposed by Ulric Neisser. Note that our levels are not hierarchical, do not build on each other, nor are they in any particular order, save that the ecological self/stimulus awareness is the most basic, and the conceptual self/meta-self-awareness is typically the most complex.

meta-self-awareness Meta-self-awareness is one of the levels of computational self-awareness we propose, indeed the highest one in our framework. It refers to the capability of a system to be aware of its own self-awareness. This can be very useful, since it means a system has knowledge, obtained at run time, about its own self-awareness processes, including, for example, how effective its learning is at present, or how much resource is being spent to maintain its knowledge. Meta-self-awareness is closely related to, and permits, meta-reasoning. It is a concept inspired directly from human psychology.

private self-awareness Private self-awareness refers to a system's ability to obtain knowledge based on phenomena that are internal to itself. A system needs internal sensors to achieve this. Again, this is a notion which exists in human self-awareness theory, and also features in computational self-awareness.

public self-awareness Public self-awareness refers to a system's ability to obtain knowledge based on phenomena external to itself. Such knowledge depends on how the system itself senses/observes/measures aspects of the environment it is situated in, and includes knowledge of its situation and context, as well as (potential) impact and role within its environment. This is a notion which exists in human self-awareness theory, and also features in computational self-awareness.

scope of self-awareness The scope of self-awareness refers to the domain of phenomena able to be sensed and modelled by the self in question. For a system which is only privately self-aware, the scope may be the same as the span (i.e., it has no perception of its environment). For a system which has some private and some public self-awareness, the scope would be larger than the span, and include external social or physical aspects of the environment. The term scope can be useful to avoid having to use the word "level" to mean multiple things simultaneously in a passage of text.

self-aware system We do not formally define this in the book, however we generally consider a self-aware system to be one which (at least) obtains and maintains knowledge relating to itself (including its perspective of its environment), without external control.

self-awareness capability When a particular level of self-awareness is present in a system, we refer to this as the system having that particular self-awareness capability. For example, a node may have a time-awareness capability, indicating that

it implements the time-awareness level. Levels may be realised in different ways simultaneously in the same system, meaning that, for example, a system may have several time-awareness capabilities.

self-explanation Another form of self-expression when based on self-awareness, self-explanation is the ability of a system to explain/justify its behaviour to an entity on the outside (such as a user or another system).

self-expression Self-expression is, in the general sense, behaviour based on self-awareness. It may include a wide range of different actions, enacted through a system's actuators, including self-adaptation, self-explanation, or just normal system behaviour. Self-expression can also be considered as a property of a collective, since a collective's behaviour can also be based on collective self-awareness. Examples of this might include the adaptive behaviour of a flock of birds in response to an external (to the flock) stimulus.

self-expression capability As with self-awareness capabilities, self-expression capabilities refer to the presence of an implementation of self-expression in a system. For example, a system which adapts its parameters in response to its goal-awareness, would have a self-expression capability. Again, multiple self-expression capabilities may be present simultaneously.

self-knowledge Self-knowledge is a general term for knowledge (usually held in a learnt model) concerning the system itself, which typically is produced as part of a self-awareness process. Note that this can include objective self-knowledge (i.e., about the system as an object in the world, how it interacts with others, how its internal state changes, etc.) and also subjective self-knowledge (i.e., about its experiences, sensor data, changing context, etc.).

self-optimisation Self-optimisation is a form of self-expression; self-optimisation is the ability of a system to optimise itself by improving metrics such as performance or power consumption.

span of self-awareness We use this term to refer to the domain of the subject of the self-awareness, i.e., it is the answer to the question: who is the self here? For example, if a single agent is self-aware, then the span is the agent. If we are considering the collective self-awareness of a network of smart sensors, then the span would be the network. The term span can be useful to avoid having to use the word "level" to mean multiple things simultaneously in a passage of text.

Engineering Self-aware Systems

(architectural) pattern We produced eight architectural patterns, which are derived from the reference architecture and describe how various capabilities (such as levels of self-awareness, etc.) can be included or excluded as appropriate to the application need.

methodology for engineering self-aware systems We developed a methodology for engineering self-aware systems, based on the reference architecture and the derived architectural patterns.

primitive A primitive is a particular block in the reference architecture, representing, for example, a level of self-awareness, self-expression and a sensor. They are instantiated for particular applications.

reference architecture We developed a reference architecture which captures the core aspects of computational self-awareness. The aim is to provide a common, principled basis on which researchers and practitioners can structure their work. We have argued that the psychological foundations, while not strictly necessary, can provide a means of channelling a wide range of ideas, which would perhaps otherwise not have occurred to engineers, acting to inspire the design of future computing systems. The architecture can also be used as a template for identifying common ways of implementing self-awareness capabilities. Different implementations of the same capability can thereby be compared and evaluated. Further, we have derived a set of architectural patterns from the reference architecture.

(self-aware) node We use the term self-aware node to refer to various types of system that are self-aware, e.g., an agent, a robot and a camera. Agent is an alternative term, but node can be used when not wanting to be specific about a particular system being an agent. We also claim that self-aware collectives (see next entry) can be viewed as self-aware nodes, at a higher level of abstraction. A node may or may not correspond to a physical system—this is not a requirement, but it may often make sense to make it correspond.

tactic/algorithm/technique A tactic is a particular instantiation of a primitive in the reference architecture, typically referred to as a particular algorithm, technique, etc. These are application specific. Multiple tactics may be suitable for a particular primitive, and some tactics may implement multiple primitives simultaneously.

Related Approaches

autonomic (computing) Autonomic computing is a vision originally pioneered by IBM, of engineered systems which manage themselves. This self-management is stated to include: self-configuration, self-optimisation, self-healing and self-protection. The aim is to reduce the need for human involvement in the management of complex computing systems. Some autonomic computing literature mentions the need for self-awareness as a characteristic to support self-management, though the literature on autonomic computing does not significantly expand on this. (Not to be confused with autonomous.)

autonomous (system) Autonomy is a broad notion with much disagreement surrounding it. However, in general, an autonomous system is one which acts without any external direction. Examples include robots, vehicles and software agents. In many cases, this ability to make decisions is based on a method of decision making pre-programmed into the system, in other cases it is learnt online at run time. The

types of systems we are concerned with in this book are ones which would typically be considered to be autonomous to a greater or lesser extent. (Not to be confused with autonomic.)

metacognition/metareasoning Metareasoning is reasoning about reasoning, and has been the topic of a significant amount of research primarily in the US, where it has been primarily led by DARPA. Metareasoning relies on meta-self-awareness, and again the metareasoning community has discussed self-awareness as being important, but not expanded on the notion significantly.

organic computing This is a vision from a long-running (primarily) German research project to create "life-like" engineered systems, in which self-organising emergent behaviour is controlled (by an observer/controller component), to ensure desirability in the self-organisation. The Organic Computing literature also mentioned self-awareness as beneficial, but again does not expand on this significantly.

General Terms

adaptability In high level terms, this is similar to adaptivity, but describes a system's potential for adaptation, rather than actual realised adaptivity.

adaptivity In high level terms, this concerns the amount to which a system adapts, e.g., in the presence of a changing environment, or as a result of its learning.

collective We use the term collective to refer to various types of distributed systems, typically without central control. Examples include swarms, systems-of-systems, populations, multi-agent systems, interwoven systems, etc. The term can be used when there is a need to talk generally of these types of systems, without restricting the discussion to a specific one.

learnt model A learnt model is a model which has been induced through a process of (typically online) learning, based on data from sensors and other existing models. Learnt models hold the conceptual knowledge a self-aware system has concerning itself, its interactions, history, expectations, goals, etc.

model We use the term model in a very general way, to refer to a conceptual representation of some knowledge, typically obtained through sensors. A model could simply be a direct representation of some data, or could be abstractions of that data, or further data synthesised from sensory input.

online learning Online learning is the process of learning a model from data on an ongoing basis. Typically, not all data is available in advance (e.g., it arrives in a streaming fashion from sensors), and the concept being learnt may change over time (i.e., concept drift). In online learning, models are often used (e.g., through self-expression in this case) before learning "completes", if indeed it ever does. Hence most online learning algorithms also need to be anytime algorithms, implying that models are used and improved continuously as time goes by.

self-adaptive system A system which adapts (typically its behaviour) in response to external or internal changes, but without external control. We have argued that

self-awareness is an enabling property for effective self-adaptation. When self-adaptation behaviour is based on self-awareness, it is a form of self-expression.

self-organising system A system which changes its organisation (e.g., its structure, architecture, topology), without external control.