Self-awareness in Autonomic Systems
Concepts and Challenges

Franco Zambonelli
Università di Modena e Reggio Emilia

Outline

- Why?
  - Self-awareness as a key enabler to autonomicity

- What?
  - Concepts and implications of self-awareness

  - Challenges in modelling, engineering and building self-aware autonomic systems
Why Autonomicity?

- Increasing complexity of ICT systems
  - Mobility and ephemerality of nodes, variability of requests, location-based systems
  - Heterogeneity of computing devices and communication media
  - Need of reliability and 24/7 availability
  - Need of “user-centric” adaptability
- Challenging human-centered approaches to ICT systems management
  - Economically unbearable
  - Too much factors to be accounted for beings of limited rationality

Why Self-awareness?

- Autonomic approaches consider that the role of human managers can be played by ICT systems themselves
  - Having them include the capability of self-monitoring and of self-acting
  - To realize a sort of control loop
- Of course, this requires
  - The capability of understanding what’s happening to themselves
  - The capability of knowing what will happen in response to self-action
What has been done so far?

- Autonomic computing and communication researches have mostly focussed on architectural approaches
  - How to realize the autonomic control loop within ICT systems (e.g.: the MAPE-K model of IBM; the ACE-based component model of the CASCADAS project; self-* P2P networks architectures etc.)
  - What models, algorithms, tools, should be defined to promote autonomic systems (e.g.: nature-inspired algorithms, negotiation techniques, multiagent systems, etc.)
- Yet
  - The concept of “self-awareness” still requires to be properly unfolded in its various facets (most approaches so far focus only on “context awareness” for localized actions)
  - As it is the impact of “self-actions” on systems’ behaviour and complexity (mostly understudied)

What is “System”?

- What systems are we referring to? (e.g., where is self-awareness an issue?)
  - Large service/data centers
  - Internet and Telecom networks
  - Pervasive, mobile, and ad-hoc networks
  - Social networks and business ecosystems
  - System-level services vs. User-level services
- But the above distinction makes little sense
  - Systems are converging, being integrated, in need to coordinate with each other and work in an orchestrated way
- Need of general, foundational approaches, accounting for the current and future integration trends
  - Application-specific and technology-specific approaches could be of help, but would be definitely of limited impact
Becoming aware of what?

- What are the situations of which a system should become aware of?
  - E.g. what should a system know to effectively self-express?
- Hard to confine
  - We can hardly abstract away from a large number of different situations that could impact the overall behaviour of the system and the satisfaction of the stakeholders
- Nothing like traditional layering or vertical integration can work
  - Yet dealing with all this information at a once can be very challenging

But...are we talking about data?

- No, collecting data granules is not an issue (sensors’ everywhere!)
  - FRANCO’s iPhone: GPS(lat, lon); Battery Level: 8%; Available Networks: WiFi, UMTS; Temperature: 3°; Activity: watching “Gone with the Wind”
- Getting a meaning out of diverse data is, indeed
  - Franco is close to home and will get there soon, as he is not expected to watch all the movie outside with such a freeze!
  - So I (the iPhone) do not have to worry about saving battery power now by staying off-line, because Franco will soon have a chance to recharge me
- The key point is knowing what data to analyse and turning it into high-level knowledge about situations!
How?
Data management and software engineering challenges

- Extracting knowledge from data
  - There is need of discriminating among possibly overwhelming amounts of data
  - There is need of correlating data granules with each other
  - There is need of recognizing situations of interest
  - **Challenges**: data models, semantics, common-sense analysis, historical analysis, knowledge overflow, etc.

- Dealing with multi-level abstractions
  - Not all components need the same details
  - We should not forget the lessons of mainstream software engineering (abstraction! abstraction! abstraction!)
  - **Challenges**: dynamic self-expression, to enable multiple degree of awareness within the same overall model and within the same software engineering abstractions

Where?
Architectural and algorithmic challenges

- Where should the capability of becoming self-aware reside?
  - Locally to each component
  - Globally to the system

- Where should information
  - Be collected, propagated, and how
  - Reside and be exploited

- Should all of this occur
  - In a separated plane?
  - As components’ inherent behaviour?

- **Challenges**: find the proper architectural and algorithmic trade-offs between
  - Local vs global
  - Separation vs integration of concerns
**When?**

**Prediction and evolution challenges**

- **Short-term**
  - Knowing what is happening is important and can be effective to correct situations
  - Knowing what will happen is important to prevent incorrect situations to occur
  - **Challenge:** Self-aware systems should become “future-aware”

- **Long-term**
  - How is my technological context (kinds of computing devices, type of connectivity, services provided) changing?
  - How is my context of usage (social/economical) changing?
  - **Challenge:** defines models and architectures so as to be “future-proof”, ensure long-lasting effectiveness and evolvability

**Who?**

**Business models and security challenges**

- **Who can act on the system?**
  - Answer: everybody!
  - Open production models (prosumption)
  - Long tail of needs and requirements (accommodate diversity)
  - **Challenge:** how can I protect my system and its data without erecting walls yet accommodating diversity and openness to prosumers? How can the system become aware of privacy/security issues?

- **Who controls the controllers?**
  - If all autonomic behaviour is based on self-awareness
  - How can the system become aware of self-unawares?
  - **Challenge:** ensure autonomic behaviour in the self-awareness-related parts of the system, yet avoiding infinite levels of recursion (meta-meta-meta-levels)
Conclusions

- Self-awareness is a key ingredient to promote autonomic behaviour in ICT systems
- Yet, there are many foundational aspects and challenges to be addressed to effectively achieve it
  - Extracting knowledge from data
  - Dynamic self-expression of proper abstraction levels
  - Global vs. local awareness
  - Separation vs. integration of concerns
  - Prediction and evolution
  - Security and self-self-awareness
- Not clear what the best approaches could be
- Definitely, plenty of room for fascinating, foundational, visionary, research!