Welcome

Formal Methods for Autonomic Systems: a special case?

Andrew Butterfield – Lero@TCD

29th January 2010
• Talk outline:
  – What are autonomic systems?
  – Formalisation state of the art
  – Autonomy as:
    • Control Theory?
    • Hybrid Systems?
  – The SFASE and FDTAS Lero proposals
  – Conclusions
• What are autonomic systems?
  – “autonomic nervous system” [Kephart&Chess, Computer 2003]
  – “zero admin, self-managing, embedded, invisible, requires zero or little human administration” [Lightstone, EASe 2006]
  – “self-management, self-optimisation, self-healing” [Dobson, EASe 2008]
  – “Autonomic Systems form a feedback loop” [Dobson et al., ACM TAAS 2006]
• Formalising Autonomy: work to date
  – Conceptual Graphs (optimisation) [Corbett & Rouff, EASE’06]
  – MMD, Fuzzy Logic, NN (diagnosis, healing) [Dai, Hinchey & Hu, EASE’07]
  – Time-Adaptation LTL (timing, safety, liveness, stability) [Zhou et al., EASE’06]
  – Situation Logic, Markov Models (organisation) [Randles et al., EASE’08, EASE’09]
  – Formal Ontology (semantic interoperability) [Breitman & Parazolo, EASE’07]
  – jABC, Service Logic Graphs, labelled Kripke structures
    • (dynamic maze following) [Jorges et al., EASE’07]
    • (service adaptation) [Bajohr & Margaria, EASE’09]
  – JML, ORC (functionality recomposition) [Martinez & Dobson, ICSOFT’09]
  – StateCharts (healing) [Elkorobarrutia et al., EASE’08]
  – ASSL (scheduling, healing, code generation) [Vassev et al., EASE’08, EASE’09]
  – Physical Models (mobile sensor networks) [Dobson et al., EASE’09]
• Autonomy as Control Theory?
  – “interdisciplinary branch of **engineering** and **mathematics**, that deals with the behavior of **dynamical systems**” [Wikipedia]
  – Simple control loop
  – Topics
    • Stability (lack of oscillations)
    • Controllability
    • Observability
    • Common aspect: timed response to perturbation
    • Adaptive Control (adjusting parameters “On-the-fly”)
• Autonomy as Hybrid Systems?
  – Control Theory is mainly about continuous quantities varying smoothly over time.
  – Autonomic computing applications mix continuous and discrete aspects ("hybrid systems")
  – In formal methods, these manifest themselves most usually as hybrid automata
    • State machine + differential equations
    • “obvious” way to merge control theory and computation.
• The special case?
  – It seems that hybrid systems should cover autonomy, at least in principle
  – However, there does seem to be an issue to do with the degree of (conceptual) complexity in autonomic systems:
    • “We focus on generating simple, solvable problems from complex ones. Is it possible that, for modern (and especially autonomic) systems, this focus is wrong? If we accept that component interactions are complex and unpredictable, this means that they should be given priority in the design process.” [Dobson, EASe’07]
  – It seems clear that effective formalisation of autonomy will require specialised approaches.
• **Formal Methods: the “Gold Standard”**
  – What do we expect from a top-quality formal method in any case?
    • Formal languages with well-defined semantics
    • Amenable to formal (mechanical) reasoning and manipulation.
    • Supporting all phases of the development lifecycle: Requirements; Specification; Design; Coding; Verification with formal linkages at all stages
    • Tool support!
• Formal Methods: our first steps
  – We* intend to focus at first on
    • Specification, Design and Refinement between them
    • Systems whose topology changes over time
      – E.g. Mobile wireless sensor networks
    • Looking at top-down approaches
      – Avoiding “implementation bias”
    • Starting point ?: Linear Hybrid Automata
      – Goal: consider dynamic systems whose topology can change
    • Starting point ?: Hybrid I/O Automata [Lynch et al., IC 2003]
      – Comes with a notion of “trace” refinement

* (members of FSE and AUS in Lero)
• Our initial application focus will be on mobile sensor networks
  – E.g. The water-pollution sensor network [Dobson et al., EASe’09]
    • Sensors move around and move towards and track phenomena of interest
    • Some sensors may die, others may be added
    • Needs models of physical environment
    • Needs models of (logical) interconnectivity of sensors
    • Physical and logical topology are distinct, but related
• Modelling Mobile Wireless Sensors
  – Physical Aspects of a Sensor (location, battery power, etc.)
  – Computation Aspects of a Sensor (onboard data, algorithms/behaviour, ..)
  – Global Physical Environment (where it all happens)
  – Networking Aspects of all the Sensors together (connectivity, QOS, ...
• A Model Proposal
  – We will sketch out how a mobile sensor network model might look.
  – We assume every sensor has a unique identifier
  – As sensors come and go, we shall assume the presence of a sensor ‘generator’, and that a sensor can enter a ‘dead’ state
  – We envisage the model being formulated in some form of process calculus with hybrid semantics.
• The model has two aspects
  – Local – information regarding an individual sensor
  – Global – information about the sensor relationship to other sensors and the environment

  – Actual – information about the true physical state of affairs
    • Recorded as functions of time
  – Perceived – what sensor(s) believe to be true.
    • Recorded as automata state variables
• Correctness requires us to model bad things
  – In order to say that nothing bad happens, a formalism has to be able to model “badness”
  – A major problem for our sensor networks that can arise is the sensors (collective) knowledge about their environment is faulty
  – Hence we need to be able to model both reality and perception.
• Physical Attributes (Local)
  – Sensor Location
  – Battery Power
  – Sensor Readings
  – Motor Power Level
  – Radio Power
  – Etc..
• Computational Attributes (Local)
  – Sensor alive/dead
  – Perceived Location, Sensor Values
  – Logged Data in Memory
  – Wireless Connections
  – “Algorithmic” State
• Physical Attributes (Global)
  – Shape of local environment
  – Physical attributes of environment
    • Function from location and time to attribute
    • E.g Water Depth, flow direction, pollution levels..

• The actions of the sensors may feedback into the environment
• Computational Attributes (Global)
  – Connectivity of Nodes
  – Collective perception of System
  – Effectively the result of combining all computational nodes in parallel
  – Nodes close enough can synchronise (pass messages)
• Each Sensor is a hybrid automata
  – State records actual and perceived physical data
• Physical Environment:
  – Location × Time → PhysParam
• Global Connectivity
  – Synthesised from the sensor actual data and physical environment
  – Determines presence of communication channels and which sensors can access each channel
• Dynamic Process Algebra with Feedback
  – We have described a model where the top-level topology (which sensors are link to which) is a function of physical state
  – We have an intricate feedback loop:
    • Sensors initiate behaviour (movement-switching radio on-off) based on their perceived state
    • Global physical state changes as a result (also of its own dynamics)
    • Global connectivity changes as a result.
• Is this approach feasible?
• How can complexity be managed?
• Should we start with simpler problems?
• We have described an implementation, a design!
• What is the specification?
• What are the requirements?
• Modelling the Requirements
  – We want to capture the intended behaviour of the sensor network at a high level of abstraction
    • To watch out for abnormal events (e.g. Pollution)
    • To co-operate to establish extent of abnormality
    • To trace it back to its source
  – Currently unclear how to formalise such requirements!
• Conclusions
  – Autonomic Systems are a special case for formalisation
    • Considerable complexity in system state, dynamics and feedback.
  – Hybrid Automata looks like a reasonable starting point
  – A (strawman?) model has been proposed
    • Of considerable complexity!
  – Specification looks like being hard to do.
  – A worthy challenge!
Thank you!