Next-generation Autonomous Systems –
In Search of a Foundation

Joseph Sifakis
Verimag and SUSTech

SAINT Workshop on Self-aware
and Autonomous Systems

Shenzhen, May 7-8, 2019
The IoT Vision

The IoT allows objects to be sensed or controlled remotely across a network infrastructure, achieving more direct integration of the physical world into computer-based systems, and resulting in improved efficiency and predictability.
The IoT Vision – Next-generation autonomous systems

The Internet of Things

**Industrial IoT**
- Autonomous transport systems
- Industry 4.0
- Smart grids

**Human IoT**
- People’s explicit or arbitrary actions dynamically trigger control sequences or rule changes
- Intelligent services
- Semantic web

Rules can be changed, but human-driven changes are external to normal behavior
Next-generation autonomous systems

- Emerge from the needs to further automate existing complex organizations by progressive and incremental replacement of human agents by autonomous agents.
- Differ from ordinary autonomous systems which consist of purely non-embodied software working in digital environments and specialized to the solution of a limited number of algorithmically hard problems e.g. multi-agent systems and autonomic computing systems.

Main characteristics

- Exhibit “broad intelligence” being able to manage a dynamically changing set of potentially conflicting goals – this reflects the trend of transitioning from “narrow” or “weak” AI to “strong” or “general” AI.
- Complex and unpredictable environments:
  - increased role of cyber physical environments with an emphasis on the physical aspects
  - increased mobility, dynamism and geographical distribution
- Need for harmonious, in particular safe and secure collaboration with humans, and other forms of life e.g. “symbiotic” autonomy.
An autonomous system involves two different types of components, **agents** and **objects**, operating in a common **environment** so that their coordinated collective behavior meets some global goals.

- **An agent** is a reactive system (controller) interacting with components of its environment so that specific goals are met; it can monitor objects and from their environment and change their states and can coordinate its actions with other agents.

- **An object** is a physical or virtual component whose behavior can be controlled by system agents i.e. it is integrated as such when the system is designed.

- The **environment** consists of the elements of the physical and virtual infrastructure of the system that are used for the coordination between components (agents and objects) e.g. geographic coordinates to determine connectivity relationships, available communication infrastructure, devices for observability/controllability of objects.

Note that

- A component may be agent or object depending on its role in the system.
- It is an interesting question indeed how are related system and agent goals.
Next-generation autonomous systems – Basic Concepts

SYSTEM EXTERNAL ENVIRONMENT

SYSTEM

SYSTEM = Agents + Objets + System_Environment
Agents = Agent1 + Agent2
Objects = Traffic_light + Pedestrian + Human_Driven_car
System_Environment = (External_Envnt1 + External_Envnt2) x (Internal_Envnt1 + Internal_Envnt2)
Criticality goals for next-generation autonomous systems cannot be achieved under the current state of the art:

- Poor trustworthiness of infrastructures and systems e.g. impossibility to guarantee safety and security;
- Impossibility to guarantee response times in communication thus timeliness which is essential for autonomous reactive systems;
- Integration of mixed-criticality systems is hard to achieve because critical systems and best-effort systems are developed following two completely different and diverging design paradigms;

New practices emerge:

- Extensive use of learning-enabled components breaking with the traditional critical systems engineering practice – end-to-end AI-based solutions;
- In contrast with the current systems engineering practice (*), critical software is customized by updates – Tesla cars software may be updated on a monthly basis.

(*) An aircraft is certified as a product that cannot be modified including all its components even HW – aircraft makers purchase and store an advance supply of the microprocessors that will run the software, sufficient to last for the estimated 50 year production!
Next-generation autonomous systems – Facing the challenge

Systems Engineering comes to a turning point moving from small size centralized non-evolvable automated systems to next-generation autonomous systems.

- We need a general reference semantic model that could be a basis for evaluating system autonomy - *Not just a list of “self”-prefixed terms e.g. as Self-healing, Self-optimized, Self-protected, Self-aware, Self-organized, etc.*

- What are the technical solutions for enhancing a system’s autonomy? 
  For each enhancement, what are the implied technical difficulties and risks?

- There is a strong and urgent need to lay out a common engineering foundation for the development of next-generation autonomous systems. Essential issues to be addressed:

1. integration of model-based and data-driven techniques in “hybrid” design flows allowing to determine trade offs between trustworthiness and performance;

2. means for faithful modeling and simulation of a system in its physical environment (which includes humans);

3. combine empirical and proof-based validation for assessing trustworthiness and performance – open the way for new standards.
The Concept of Autonomy

In Search of a Foundation

- “Hybrid” design flows
- Modeling and Simulation
- Validation

Complexity Issues

- Autonomic Complexity
- Design Complexity

Discussion
Each system consists of agents acting as controllers on their environment and pursuing individual goals so that the collective behavior meets the system global goals.
The Concept of Autonomy – Meeting Goals

Given a set of goals and the model of an environment to be controlled, there are methods for computing plans enforcing the satisfaction of the goals.

**GOALS**
- Never reach Bad
- Eventually reach Target

**ENVIRONMENT MODEL**
A (possibly infinite) state graph with controllable (green) and uncontrollable (red) actions

**SYNTHESIS**
(Semi-algorithm)

**PLAN**
A (possibly infinite) tree with alternating controllable and uncontrollable actions
<table>
<thead>
<tr>
<th>Environment</th>
<th>Stimuli</th>
<th>Meeting Goals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thermostat</td>
<td>Room + Heating/cooling device</td>
<td>Explicit controller</td>
</tr>
<tr>
<td>Shuttle</td>
<td>Cars + Passengers+ equipment</td>
<td>Explicit controller + on line adaptation</td>
</tr>
<tr>
<td>Chess robot</td>
<td>Chess board + pawns</td>
<td>On-line planning+ stored knowledge</td>
</tr>
<tr>
<td>Soccer robot</td>
<td>Regions in the field + Players + Ball</td>
<td>On-line planning+ stored/generated knowledge</td>
</tr>
<tr>
<td>Robocar</td>
<td>Vehicles/obstacles + Road/communication</td>
<td>On-line planning+ stored/generated knowledge</td>
</tr>
</tbody>
</table>
The Concept of Autonomy – Agent Model

Knowledge Repository

- Knowledge generation

Self-awareness
- Knowledge application

Self-adaptation

Knowledge application

Autonomous Agent

Situation awareness

Perception

Reflexion
- External Environment model
- Internal Environment model

Internal Sensors

Internal Environment

Internal Actuators

External Sensors

External Environment

External Actuators

Adaptive Decision

Goal management
- Planning

Sensory information

Commands
The Concept of Autonomy – Agent Model

Autonomy is the capacity of an agent to achieve a set of coordinated goals by its own means (without human intervention) adapting to environment variations. It combines five complementary aspects:

- **Perception** e.g. interpretation of stimuli, removing ambiguity from complex input data and determining relevant information;
- **Reflection** e.g. building/updating a faithful environment run-time model from which strategies meeting the goals can be computed;
- **Goal management** e.g. choosing among possible goals the most appropriate ones for a given configuration of the environment model;
- **Planning** to achieve a particular goal;
- **Self-awareness/adaptation** e.g. the ability to create new situational knowledge and new goals through learning and reasoning

Insights on

- Automation vs. Autonomy;
- Human-assisted vs. Machine Empowered autonomy
<table>
<thead>
<tr>
<th>Level</th>
<th>SAE AUTONOMY LEVELS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level 0</td>
<td>No automation</td>
</tr>
<tr>
<td>Level 1</td>
<td>Driver assistance required (“hands on”)</td>
</tr>
<tr>
<td></td>
<td>The driver still needs to maintain full situational awareness and control of the</td>
</tr>
<tr>
<td></td>
<td>vehicle e.g. cruise control.</td>
</tr>
<tr>
<td>Level 2</td>
<td>Partial automation options available (“hands off’)</td>
</tr>
<tr>
<td></td>
<td>Autopilot manages both speed and steering under certain conditions, e.g. highway</td>
</tr>
<tr>
<td></td>
<td>driving.</td>
</tr>
<tr>
<td>Level 3</td>
<td>Conditional Automation (“eyes off”)</td>
</tr>
<tr>
<td></td>
<td>The car, rather than the driver, takes over actively monitoring the environment</td>
</tr>
<tr>
<td></td>
<td>when the system is engaged. However, human drivers must be prepared to respond to</td>
</tr>
<tr>
<td></td>
<td>a &quot;request to intervene&quot;</td>
</tr>
<tr>
<td>Level 4</td>
<td>High automation (“mind off”)</td>
</tr>
<tr>
<td></td>
<td>Self driving is supported only in limited areas (geofenced) or under special</td>
</tr>
<tr>
<td></td>
<td>circumstances, like traffic jams.</td>
</tr>
<tr>
<td>Level 5</td>
<td>Full automation (“steering wheel optional”)</td>
</tr>
<tr>
<td></td>
<td>No human intervention is required e.g. a robotic taxi</td>
</tr>
</tbody>
</table>
Three factors determine the autonomy level of a system:
1) Autonomic complexity; 2) Design Complexity; 3) Degree of Trustworthiness
The Concept of Autonomy – Awareness and Adaptation

These concepts should be deepened and clarified!!
The Concept of Autonomy

In Search of a Foundation

- “Hybrid” design flows
- Modeling and Simulation
- Validation

Complexity Issues

- Autonomic Complexity
- Design Complexity

Discussion
Hybrid Design Flows – The Principle

Data-based approach (Sufficient evidence)

Model-based approach (Guarantees)

“Hybrid” approach (Guarantees + Sufficient evidence)

Run-time assurance

Deployment

Design-time

Execution Platform

Learning-enabled Agent

“Hybrid” Autonomous Agent

Automated Agent

DIR mechanisms

Run-time
Current approaches guarantee trustworthiness at design time by applying:
- a more or less exhaustive risk analysis that identifies all kind of harmful events
- techniques guaranteeing tolerance: any single harmful event leads to non-fatal states
- DIR (Detection, Isolation, Recovery) mechanisms leading from non-fatal states to trustworthy states

These approaches cannot be directly applied to autonomous systems:
- Lack of predictability and environment complexity make practically impossible identification at design time of all harmful events and corresponding DIR mechanisms
- Use of learning-enabled components

Non-Trustworthy States

Trustworthy States

Fatal States

Non-Fatal States
Pre-crash failure typology covering 99.4% of light-vehicle crashes for 5,942,000 cases.
Source: Pre-Crash Scenario Typology for Crash Avoidance Research, DOT HS 810 767, April 2017.

FDIR approaches are not anymore applicable due to overwhelming complexity!
Mobileye’s Responsibility-Sensitive Safety: Compute lower bounds of the distance between two cars that guarantee safety. ("On a Formal Model of Safe and Scalable Self-driving Cars" Shai Shalev-Shwartz, Shaked Shammah, Amnon Shashua, Mobileye, 2017)

Safe Distance Formula

\[ d_{\text{min}} = L + T_f [v_r - v_f + \rho (a_a + a_b)] - \frac{\rho^2 a_b}{2} + \frac{(T_r - T_f)(v_r + \rho a_a - (T_f - \rho) a_b)}{2} \]

- \( L \) is the average length of the vehicles
- \( \rho \) is the response time of the rear vehicle
- \( v_r, v_f \) are the velocities of the rear/front vehicles
- \( a_a, a_b \) are the maximal acceleration/braking of the vehicles
- \( T_f \) is the time for the front car to reach a full stop if it would apply maximal braking
- \( T_r \) is the time for the rear car to reach a full stop if it would apply maximal acceleration during the response time, and from there on maximal braking

See also “The Safety Force Field” David Nistér, Hon-Leung Lee, Julia Ng, Yizhou Wang, Nvidia White Paper, March 2019
The general problem:

1. An agent provides **critical** services and possibly some **non-critical** services.

2. The agent uses a variable amount of free resources $F$ (measured in space, time, memory, energy, etc.) such that $F_{\text{min}} \leq F$ and $|\frac{\partial^2 F}{\partial t^2}| \leq a_{\text{max}}$

   - $F_{\text{min}}$ is sufficient for the system to ensure the critical services
   - Critical services should be absolutely ensured (safety)
   - The rest of the available resources should be used in the best possible manner to ensure non critical services (performance).

- Safety cannot be dissociated from performance e.g. overtaking on a two lane road

- The problem needs to be solved for a humongous number of configurations:
  - use learning-enabled techniques to recognize types of configurations
  - for each identified type, apply a model-based protocol
- The Concept of Autonomy
- In Search of a Foundation
  - “Hybrid” design flows
  - Modeling and Simulation
    - Validation
- Complexity Issues
  - Autonomic Complexity
  - Design Complexity
- Discussion
Most simulation systems use ad-hoc techniques coupling an autonomous monolithic agent to game SW. We need
- Building scenarios that capture behavior corner cases and high risk situations
- Building environment models incrementally and compositionally
- Different levels of abstraction from fine grain simulation of cyber physical components to high level simulation e.g. results reported by Waymo: 27,000 cars running 24/7, 10 million miles simulated per day, >7 Billion miles of simulation.

A component-based modeling framework for autonomous systems should integrate the following features:

1. Libraries of component types for both agents and objects, as well as libraries of architecture patterns and protocols;
2. Expressive component coordination primitives supporting parametric description and various types of dynamism such as component creation/deletion and mobility;
3. Self-organization by supporting multi-mode coordination e.g. a component can live in many different “worlds” and migrate according to its pursued goals.
4. Knowledge management and application for situational awareness and generation of new goals accordingly.
Modeling and Simulation – Autonomous System Architecture

DR-BIP (Dynamic Reconfigurable BIP)

- A system is a set of (architecture) motifs
- A motif is a coordination mode consisting of
  - A set of components, instances of types of agents or objects
  - A map that is a graph (N,E) used to describe relations between components e.g. geographical, organizational, etc.
  - An address function @ mapping components into nodes of the map
- Interaction rules: define interactions (atomic multiparty synchronization) between components
- Configuration rules:
  - Mobility of components (change of @)
  - Creation/deletion of components
  - Dynamic change of the map

The meaning of systems models is defined using operational semantics
Interaction rule:
for all a,a':vehicle, if \([\text{dist}(\@a,\@a')<l]\) then exchange(a.speed,a'.speed).

Mobility rule:
for all a:vehicle if \(@a=n\) and \(@^{-1}(n+1)=\text{empty}\) then \(@a:=n+1\).
Model-based Approach – Refined Agent Model

Knowledge Repository
- Agent types
- Object Types
- Map Patterns
- Declarative knowledge
- Methods
- Goals

Knowledge generation

Self-awareness
Self-adaptation

Perception

sensory information

Agent's Environment Model

Reflexion

Goal management

Planning

Environment Model

Sensors

commands

Actuators
The Concept of Autonomy

In Search of a Foundation
  - “Hybrid” design flows
  - Modeling and Simulation
  - Validation

Complexity Issues
  - Autonomic Complexity
  - Design Complexity

Discussion
In Search of a Foundation – Validation

- Machine learning techniques cannot be formally verified as they are not developed based on formal goals e.g. specifying how a dog looks different from a cat - instead, we are showing a whole bunch of pictures so they can learn just like a human learns the differences between a cat and a dog.

- Pushing model-based validation techniques to the limits

- Increasing confidence in ML-models which remain mostly “black boxes”
  - Metamorphic testing: \( \exists \phi_1, \phi_2 \text{ if } y = f(x) \text{ then } \phi_2(y) \approx f(\phi_1(x)) \)
  - Determining reference models (oracles) i.e. interpretability, explainability, “causal modeling”

- Combining proof-based and empirical validation techniques
Formalization of goals for autonomous systems is extremely hard e.g. “behavioral competencies” for self-driving cars (California PATH)

1. Detect and Respond to Speed Limit Changes and Speed Advisories
2. Perform High-Speed Merge (Highway)
3. Perform Low-Speed Merge
4. Move Out of the Travel Lane and Park (e.g., to the Shoulder for Minimal Risk)
5. Detect and Respond to Encroaching Oncoming Vehicles
6. Detect Passing and No Passing Zones and Perform Passing Maneuvers
7. Perform Car Following (including Stop and Go)
8. Detect and Respond to Stopped Vehicles
9. Detect and Respond to Lane Changes
10. Detect and Respond to Static Obstacles in the Path of the Vehicle
11. Detect Traffic Signals and Stop/Yield Signs
12. Respond to Traffic Signals and Stop/Yield Signs
13. Navigate Intersections and Perform Turns
14. Navigate Roundabouts
15. Navigate a Parking Lot and Locate Spaces
16. Detect and Respond to Access Restrictions (One-Way, No Turn, Ramps, etc.)
17. Detect and Respond to Work Zones and People Directing Traffic in Unplanned or Planned Events
18. Make Appropriate Right-of-Way Decisions
19. Follow Local and State Driving Laws
20. Follow Police/First Responder Controlling Traffic (Overriding or Acting as Traffic Control Device)
21. Follow Construction Zone Workers Controlling Traffic Patterns (Slow/Stop Sign Holders).
22. Respond to Citizens Directing Traffic After a Crash
23. Detect and Respond to Temporary Traffic Control Devices
24. Detect and Respond to Emergency Vehicles
25. Yield for Law Enforcement, EMT, Fire, and Other Emergency Vehicles at Intersections, Junctions, and Other Traffic Controlled Situations
26. Yield to Pedestrians and Bicyclists at Intersections and Crosswalks
27. Provide Safe Distance from Vehicles, Pedestrians, Bicyclists on Side of the Road
28. Detect/Respond to Detours and/or Other Temporary Changes in Traffic Patterns
Rigorous System Design – Model-based Validation

- Formal verification
  - is applicable when goals that can be explicitly formalized as requirements
  - Is tractable for moderate model complexity - only monolithic verification techniques of finite state systems can be automated;
  - Is not enough! Autonomy is about controller synthesis under both safety and optimization constraints;
  - A more natural approach is to achieve correctness by design.

The V-model, Systems Engineering Process recommended by Safety Standards such as ISO26262

1. assumes that all the system requirements are initially known, can be clearly formulated and understood.

2. assumes that system development is top-down from a set of requirements. Nonetheless, systems are never designed from scratch; they are built by incrementally modifying existing systems and component reuse.

3. considers that global system requirements can be broken down into requirements satisfied by system components.
The Concept of Autonomy

In Search of a Foundation
  - “Hybrid” design flows
  - Modeling and Simulation
  - Validation

Complexity Issues
  - Autonomic Complexity
  - Design Complexity

Discussion
Complexity Issues

Which factors characterize the difficulty to build autonomous systems?

- **Autonomic complexity**: accounts for the specific difficulty to deal with the key five aspects of system autonomy.

- **Design complexity**: characterizes the difficulty to build a system out of components factored into
  - **Architecture complexity**: the difficulty to coordinate system agents to achieve global system goals
  - **Agent reactive complexity**: characterizes the intricacy of the interaction between an agent and its environment - independent from space complexity or time complexity

- **Implementation complexity**: characterizes the difficulty to realize the model coordination mechanisms for a given mapping of agents into the available computing/communication infrastructure.
Complexity Issues – Autonomic Complexity

- **Complexity of perception** the difficulty to interpret stimuli (cope with ambiguity, vagueness) provided by the environment and to timely generate corresponding inputs for the agent environment model.

- **Lack of observability/controllability** which implies on line computation of plans from a partial model of the agent’s environment.

- **Complexity of goal management**: compute a maximal subset of compatible goals characterizing a strategy for which a consistent plan is generated.

- **Complexity of planning**: depends on the type of goals and the complexity of the agent’s environment model.

- **Self* Complexity** which is directly related to uncertainty about the agent’s environment.
  Sources of uncertainty are multiple, including time-varying load, dynamic change due to mobility, bursty events, and most critical events such as failures and attacks.
  Note that reduced observability is a source of uncertainty. Nonetheless, uncertainty is not completely resolved by simply enhancing observability.
The Concept of Autonomy

In Search of a Foundation
  - “Hybrid” design flows
  - Modeling and Simulation
  - Validation

Complexity Issues
  - Autonomic Complexity
  - Design Complexity

Discussion
Complexity Issues – Design Complexity

ARCHITECTURAL COMPLEXITY

- Static
- Parametric
- Dynamic
- Mobile
- Self-organizing

AGENT REACTIVE COMPLEXITY

- Transformational
- Streaming
- Embedded CyberPhy
Reactive complexity characterizes the intricacy of the interaction between an agent and its environment. It is independent from space complexity or time complexity measuring the quantity of computational resources needed by the agent.
Architectural Complexity

How much involved is the coordination between components?

Static Architecture: Multiprocessor System

Parametric Architecture: Ring Architecture

Dynamic Architecture: Distributed System

Mobile Architecture: Mobile phones

Self-organizing Architecture: Swarm robots
Design Complexity: Reactive × Architectural Complexity

AGENT REACTIVE COMPLEXITY
Transformational
Embedded
CyberPhy

ARCHITECTURAL COMPLEXITY
Static             parametric                    dynamic                 mobile        self-organizing

SERVICES
Multimedia
Connected Medical Devices
Data Analytics
Mobile Services

SYSTEMS
Active safety
Agricultural Robot
Smart Grid
Amazon Drones
Swarm Robots

Amazon
Google Cars
Drones
Agricultural Robot
The Concept of Autonomy

In Search of a Foundation
- “Hybrid” design flows
- Modeling and Simulation
- Validation

Complexity Issues
- Autonomic Complexity
- Design Complexity

Discussion
The trend for autonomous systems in the framework of IoT renders obsolete current critical systems engineering principles and standards such as such as ISO26262 and DO178B, that require **conclusive evidence** that the system can cope with any type of harmful event.

- they cannot handle machine learning software;
- they cannot handle design flows for autonomous systems – they give a system credit for a human assistant ultimately being responsible for safety.
- they require guarantees at design time and stringent predictability that are impossible to provide IoT autonomous systems.

Consequently, there is no Independent safety certification for autonomous systems! Automotive and medical products are self-certified by their manufacturers according to guidelines that determine how to provide **sufficient evidence** that the developed system is reliable enough.

Two important questions:

- Quantifying trust in data-based approaches (statistically)
- Is it possible to develop rigorous design methodologies and associated trustworthiness assessment techniques as well as standards for third party certification?
Discussion – The Way Forward

- We provide a technical characterization of autonomy as the combination of five basic and independent features that contrasts with the so many self-* approaches involving a large number of poorly understood “self”-prefixed terms.

- We show that autonomy should be associated with functionality and not with specific techniques – while ML is essential it is not only way to build perceptors and adaptive controllers.

- We consider autonomy is a kind of broad intelligence. It is not just decision automation even if this requires the computation of strategies with exploding complexity.

- There are big differences between an autonomous vehicle and a game playing robot (static situational awareness, rules of the game well-understood).

- We urgently need a common engineering foundation for the development of next-generation autonomous systems.

Building trustworthy next-generation autonomous systems goes far beyond the current AI challenge.
Thank You