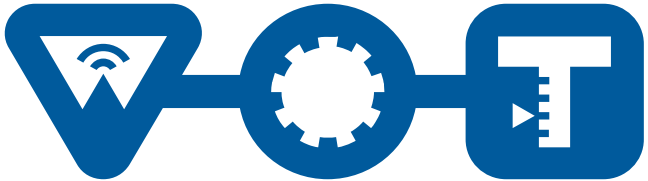
IntellioT



University of St.Gallen

Simon Mayer, Jérémy Lemée, Andrei Ciortea,
Danai Vachtsevanou, Samuele Burattini

Interactions Research Group, https://interactions.ics.unisg.ch

Talk Abstract

In the **IntellIoT Project** (https://intelliot.eu/), we used W3C WoT TD for **integrating Web-enabled industrial devices and Autonomous Agents within Multi-agent Systems**

We successfully deployed two use cases: **Autonomous Agriculture** and **Industrial Manufacturing**

Drive **Interoperability!**

Enable **Decoupling!**

Enable **Autonomy!**

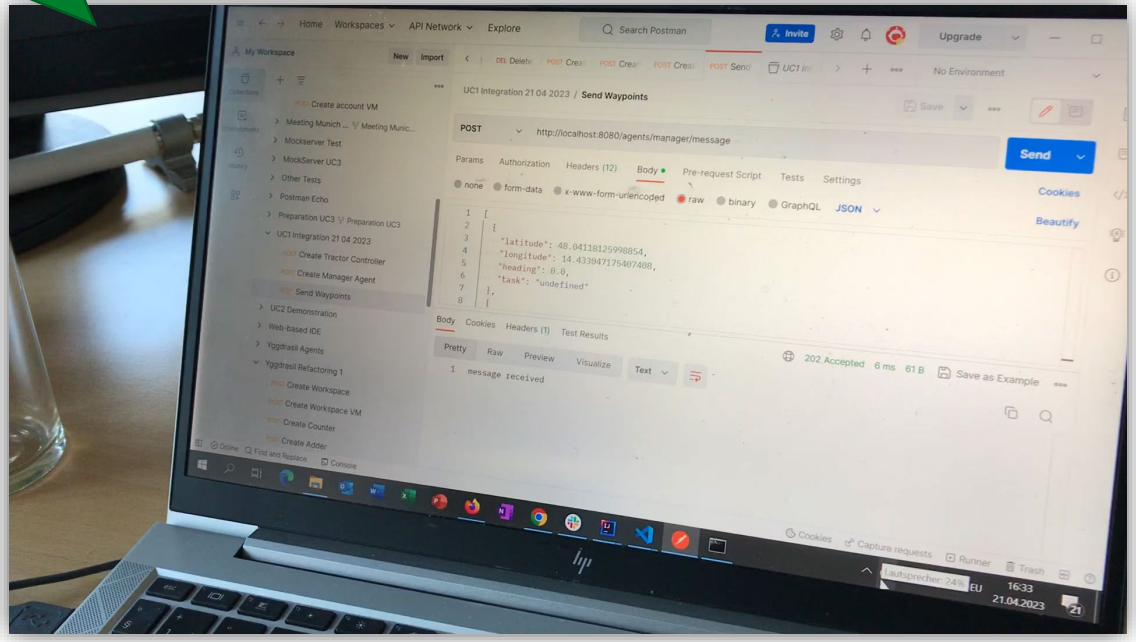
An **autonomous agent** drives a tractor **through W3C WoT TD**

Things expose WoT TDs (tractor controller, waypoint service, AI service, journaling service, human-in-the-loop service, …)

TDs are exposed via yggdrasil, a **platform for agents and artifacts**

TDs are discovered by a **no-code development environment.** A domain expert **programs an agriculture agent** to achieve operator goals

An **operator issues goals** ("Harvest field 5") at run time using a domain-specific interface



All IntelliIoT TDs: https://github.com/Interactions-HSG/example-tds/tree/intelliot/tds
Tractor TD: https://github.com/Interactions-HSG/example-tds/blob/intelliot/tds/tractor_controller.ttl
Example TD on yggdrasil: https://yggdrasil.interactions.ics.unisg.ch/environments/61/workspaces/102
Paper on no-code Development Environment (EMAS 2022): https://emas.in.tu-clausthal.de/2022/papers/paper3.pdf
Extended abstract on architecture (AAMAS 2023): https://www.alexandria.unisg.ch/269570/

An **autonomous agent** manages a robot **through W3C WoT TD**

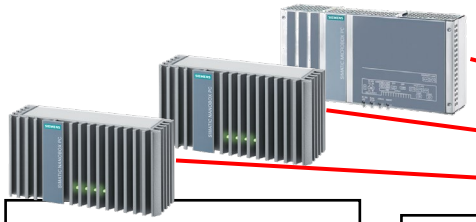
Things expose WoT TDs (robot controller, engraver, grabspot service, journaling service, human-in-the-loop service, …)

TDs are exposed via yggdrasil, a **platform for agents and artifacts**

TDs are discovered by a **no-code development environment.** A domain expert **programs a manufacturing agent** to achieve operator goals

An **operator issues goals** ("Engrave the text IntellIoT at location x/y") at run time using a domain-specific interface

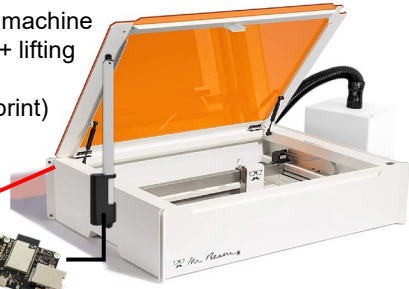
All IntellIoT TDs: https://github.com/Interactions-HSG/example-tds/tree/intelliot/tds
Robot TD: https://github.com/Interactions-HSG/example-tds/blob/intelliot/tds/uc3_robot.ttl
Example TD on yggdrasil: https://yggdrasil.interactions.ics.unisg.ch/environments/61/workspaces/102
Paper on no-code Development Environment (EMAS 2022): https://emas.in.tu-clausthal.de/2022/papers/paper3.pdf
Extended abstract on architecture (AAMAS 2023): https://www.alexandria.unisg.ch/269570/



Industrial Edge infrastructure (HW+SW) - Siemens Industrial Edge Apps (HSG, Eurecom, UOULU, AAU, TSI, Siemens)

Security Assurance Platform (HW?) - SANL+TSI

Laser Engraver machine (Laser cutter) + lifting table +SW (Octoprint)



ESP32 Ethernet Brick to control cover, start button

5G infrastructure (HW+SW) - Eurecom

TSN infrastructure (Controller + switches) - Siemens and TTTech

"Machine Controller" HW (Microbox PC + TTTech NIC)

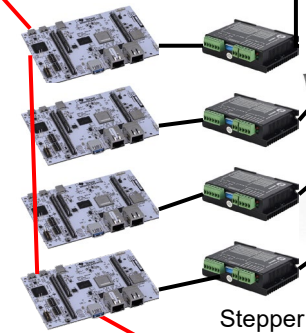
AR Infrastructure - Hololight

Industrial camera

PC UR5 Controller

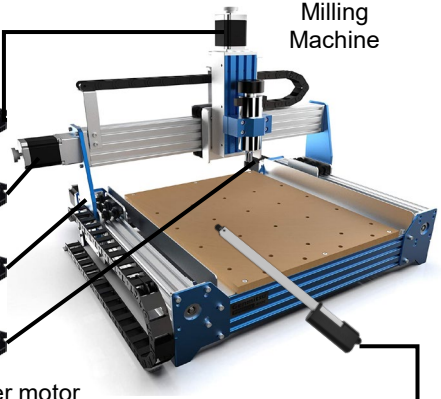
UR5 Robot

Industrial Edge Orchestrator (min 32GB RAM)



Eval boards for controlling 4 motors of Milling machine + SW

Stepper motor drivers



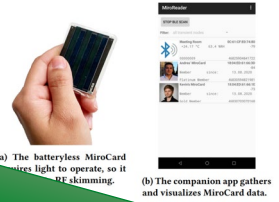
Milling Machine

actuators for fixing workpiece

ESP32 Ethernet Brick

(similar for the Agriculture use case)

Interactions of Agents with Devices are **journaled** in a DLT

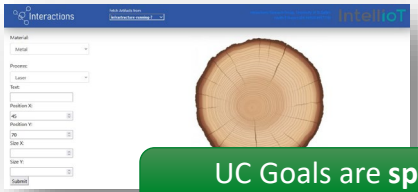


Some services are only **deployed lazily** / some device interfaces only **become bound at run time**

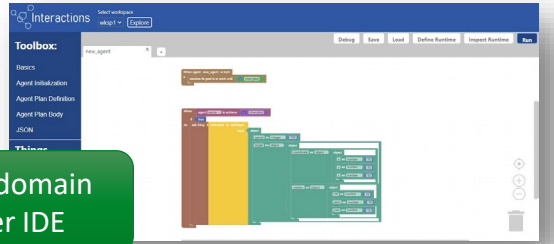
The Agents **interact with** Things/Services/Artifacts including Tiny Devices

The Agents **run** here

Devices and Services **register** here



UC Goals are **specified** through a dedicated Web front end per Use Case



The Agents are **configured** by a domain engineer in a no-code end-user IDE

Talk Abstract

IntellIoT

In the **IntellIoT Project** (https://intelliot.eu/), we used W3C WoT TD for **integrating Web-enabled industrial devices and Autonomous Agents within Multi-agent Systems**

We successfully deployed two use cases: **Autonomous Agriculture** and **Industrial Manufacturing**

I cover these core components and approaches:

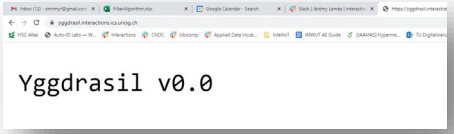
1. Management of W3C WoT TDs in the **Yggdrasil Hypermedia MAS Infrastructure**
2. W3C WoT TD-based **No-Code Development for Multi-agent Systems**
3. **W3C WoT TD Negotiation** for lazy deployment and binding of edge services
4. W3C WoT TD-based Journaling of Agent-Artifact Interactions
5. W3C WoT TDs in the context of Affordance Theory

Drive **Interoperability!**

Enable **Decoupling!**

Enable **Autonomy!**

Aspect #1: Management of W3C WoT TDs in the **Yggdrasil Hypermedia MAS Infrastructure**



https://www.alexandria.unisg.ch/256718/

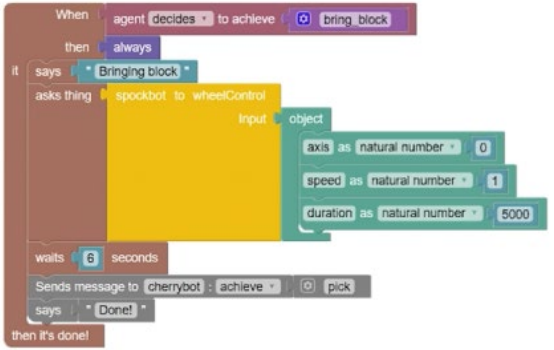
- Supports creation of **Hypermedia Multi-Agent Systems** following the JaCaMo meta-model
- Supports the execution of computational artifacts, allows agents to instantiate artifacts, exposes HTTP interfaces for interacting with the artifacts, and generates W3C WoT TDs for the instantiated artifacts
- Yggdrasil-internal artifacts are complemented by W3C WoT TDs for **external devices or services**
- Resulting hypermedia environment is represented in RDF
- Infrastructure is compatible with a search engine for the W3C WoT

https://link.springer.com/article/10.1007/s00779-020-01415-1

Yggdrasil can be a blueprint for W3C WoT TD-based systems that include Autonomous Agents

https://yggdrasil.interactions.ics.unisg.ch/environments/61

Aspect #2: W3C WoT TD-based **No-Code Development for Multi-agent Systems**

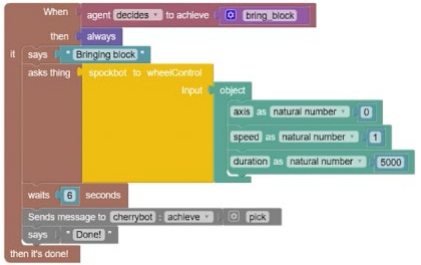


- W3C WoT TD-based **visual programming for agents** (AgentSpeak language)
- Configure and deploy Hypermedia Multi-agent Systems (extended to MAS organizations)
- Block language based on Blockly. Functional blocks are **automatically generated** from W3C WoT TDs
- **Web-based IDE fetches W3C WoT TDs from Yggdrasil and generates functional blocks**
- Configured agents are **deployed** to Yggdrasil
- Agent behaviors can be implemented against W3C WoT TD affordances while **resolving protocol binding only at run time**

This Web-based Agents IDE can be used to create agents based on W3C WoT TD

https://emas.in.tu-clausthal.de/2022/papers/paper3.pdf

Aspect #2: W3C WoT TD-based **No-Code** Development for Multi-agent Systems



Agent-Oriented Visual Programming for the Web of Things

Samuele Burattini [1] Alessandro Ricci [1] Simon Mayer [2]
Danai Vachtsevanou [2] Jeremy Lemee [2] Andrei Ciortea [2]
Angelo Croatti [1]

Dipartimento di Informatica - Scienza e Ingegneria,
Alma Mater Studiorum, Università di Bologna, Cesena Campus, Italy
samuele.burattini@studio.unibo.it, {*a.croatti—a.ricci*}*@unibo.it*

Interaction- and Communication-based Systems, Institute of Computer Science,
University of St. Gallen, Switzerland
{*andrei.ciotea—danai.vachtsevanou—jeremy.lemee—simon.mayer*}*@unisg.ch*

27th April, 2022

https://emas.in.tu-clausthal.de/2022/papers/paper3.pdf

Aspect #3: W3C WoT TD Negotiation

- Partners desire deploy services/things **as late as possible for optimization of industrial edge** ("Edge Orchestration")
- Usage of W3C WoT TD Templates (W3C WoT TD Version 1) and **W3C WoT Thing Models** (W3C WoT TD Version 1.1) for permitting late binding of edge-orchestrated services
- **IntellIoT Edge Orchestrator** emits W3C WoT Thing Models (i.e., Thing Descriptions without protocol binding)
- Agents are programmed (using the Web IDE) **against these Thing Models**
- Deployed agents request services from Edge Orchestrator. EO supplies protocol-bound W3C WoT TD

Proposal: W3C WoT TD Negotiation

- Parameters of machine-learning systems (capabilities)
- Security parameters
- **Any non-functional parameter of a service/thing can be used in this sense**

Aspect #4: W3C WoT TD-based Journaling of Agent-Artifact Interactions

- Implementation of prototype for **journaling of W3C WoT TD-based interactions** in a distributed ledger
- Purpose: Monitoring, auditing, root-cause analysis, possibly payments
- Recording of concrete exchanged messages (according to TD-binding)
- Recording of the service descriptions that are effective at run time
- In IntellIoT implemented using a distributed ledger; any other (trusted) journaling infrastructure may be used

Aspect #5: W3C WoT TDs in the context of Affordance Theory

https://www.alexandria.unisg.ch/269015/

https://hyperagents.org/

- Collaboration with the HyperAgents project
- Investigation of **Affordance Theory** in the context of the guidance of interactions of autonomous hypermedia clients
- Introduction of **signifiers as a first-class abstraction** in Web-based MAS
- Differentiation between signifiers and affordances to allow **run-time adaptation of exposed IDLs**
  - Adapt to agent goals
  - Adapt to agent abilities (including cognitive abilities)

Same functionality, but advertised…

```
...
<#sig> a hmas:Signifier ;
 hint:signifies <#close-gripper> ;
 hint:recommendsAbility [ a prs:PRSAbility ] ;
 hint:recommendsAbility [ manu:OperatorAbility ] ;
 hint:recommendsContext <#env-context>, <#prs-context> .

<#prs-context> a hint:Context; sh:targetClass hmas:Agent ;
 sh:property [ sh:path prs:hasDesire ;
  sh:minCount 1 ; sh:qualifiedMinCount 1 ;
  sh:qualifiedValueShape <#desire-shape> ] .

<#desire-shape> a sh:NodeShape ;
 sh:class manu:PickAndPlace;
 sh:property [ sh:path prs:hasInputList
    ... ] .

<#item-shape>  a sh:NodeShape ;
 sh:class manu:Item ;
 sh:property [ sh:path manu:hasLocation ;
   ... ] .

<#location-shape> a sh:NodeShape ;
 sh:class manu:Location ;
 sh:property [ sh:path manu:inRangeOf ;
  sh:minCount 1 ;
  sh:hasValue ex:leubot ] .
```

```
@prefix pddl:
<http://www.cs.yale.edu/homes/dvm/daml/pddlonto.daml#>.
...
<#sig> a hmas:Signifier ;
 hint:signifies <#close-gripper> ;
 hint:recommendsAbility [
  a strips:StripsPlanningAbility ] .

<#close-gripper> a hint:ActionSpecification;
...
 a a pddl:Action ;
 pddl:action-label "closeGripper";
 pddl:parameters [ a pddl:Param_seq ;
  rdf:_1 <#param1> ];
 pddl:precondition [
  ... ] ;
 pddl:effect [
  ... ] .

<#param1> a pddl:Param ;
 pddl:name "?gv" ;
 drs:type manu:GripperValue ;
 :hasSchema <#gripper-schema> .
```

…for Agents that are based on the Procedural Reasoning System

…for Agents that incorporate a STRIPS planner

Talk Abstract

In the **IntellIoT Project** (https://intelliot.eu/), we used W3C WoT TD for **integrating Web-enabled industrial devices and Autonomous Agents within Multi-agent Systems**

We successfully deployed two use cases: **Autonomous Agriculture** and **Industrial Manufacturing**

I cover these core components and approaches:

1. Management of W3C WoT TDs in the **Yggdrasil Hypermedia MAS Infrastructure**
2. W3C WoT TD-based **No-Code Development for Multi-agent Systems**
3. **W3C WoT TD Negotiation** for lazy deployment and binding of edge services
4. W3C WoT TD-based Journaling of Agent-Artifact Interactions
5. W3C WoT TDs in the context of Affordance Theory

Drive **Interoperability!**

Enable **Decoupling!**

Enable **Autonomy!**

New W3C Community Group on **Autonomous Agents on the Web** (https://www.w3.org/community/webagents/)

Towards World-Wide Autonomous Systems

Autonomy of different components in IoT systems is becoming **more and more relevant across domains**!

Traditionally fragmented communities (Web of Things, Web Architecture, Autonomous Agents, Multiagent Systems, Semantic Web)

Dagstuhl Seminar and new **W3C Community Group on Agents on the Web** integrates these communities

Let's join forces to drive interoperability and autonomous behavior on the Web of Things!

https://www.w3.org/community/webagents/

https://www.dagstuhl.de/en/seminars/seminar-calendar/seminar-details/23081