Fraunhofer Institute for Integrated Circuits IIS

1st International Workshop on Semantic Web on Constrained Things at ESWC 2023, 28th May 2023

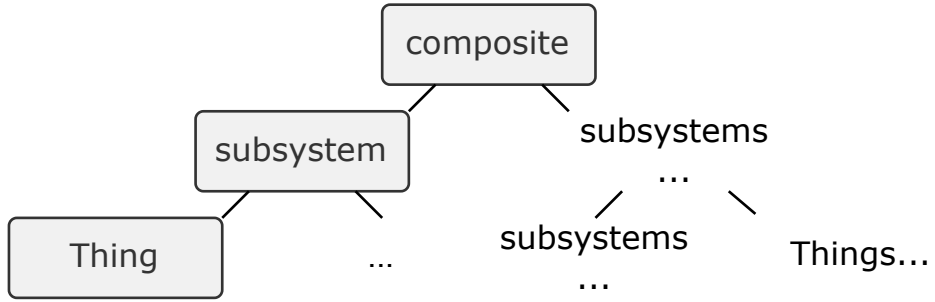
A Solid Architecture for Machine Data Exchange with Access Control

Justus Fries, Michael Freund and Andreas Harth

Motivation

A Solid Architecture for Machine Data Exchange with Access Control

- Exchange historical data of machines with stakeholders
  - Historical data: store data long-term
  - Machines: static composite Things with inherent hierarchical structure
  - Stakeholders: machine owner, vendors, customer
- Provide mechanism to interact with the composite Thing as a whole
- Enable analyzing and monitoring longitudinal data



Design Aspects

- Constrained devices: compute, data storage, energy
  - Offload processing and/or data to less constrained components
- Security and Privacy: Fine-grained, revokable access control to data of specific devices or subsystems
  - Uniquely identify stakeholders and all Things
- Attempt to ease data and system integration
  - Integrate many different Things (protocol-, vendor-agnostic)
  - Integrate and link dynamic (runtime) Thing data to static Thing data
  - Data exchange via REST API

Example Use Cases

- Static and dynamic data integrated
  - Ease data integration effort for stakeholders
- Provide stakeholders access to data of specific Things (i.e., the whole composite, subsystems, or standalone Things)
  - Able to revoke access, e.g., based on contract duration

- Machine owner can access everything
- Give customers ability to interact with machine through task queue access, machine executes tasks
- Give external maintenance worker sent by vendor access to data of a machine part

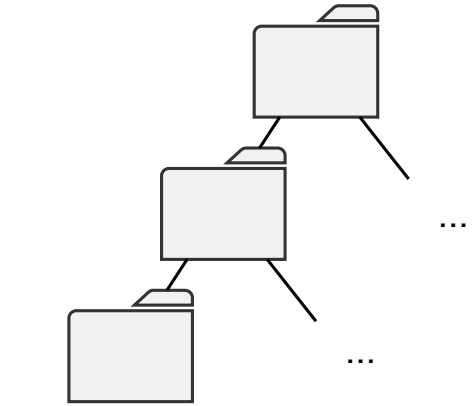
Background

Semantic Web technologies

- Web of Things (WoT) Architecture [1]
  - Thing Description (TD) [2]
    - Data linked to corresponding TDs
    - Abstract away protocol details, WoT interaction affordances

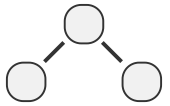
- Solid (Social Linked Data) Pod (Personal Online Data store)
  - RDF data store that resembles a file system
  - Linked Data Platform (LDP) [3]
    - Directory ≈ LDP container
    - File ≈ LDP resource
  - Exposes file system as REST API with controlled access to LDP containers and LDP resources
  - Uniquely identify stakeholders with WebID

- How exactly data is linked is up to the implementation

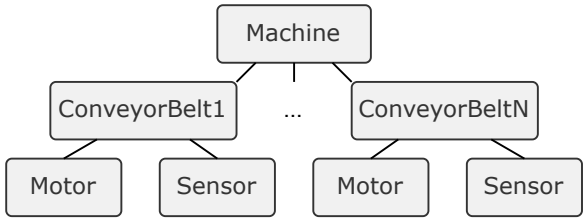


Assumptions And Composite Thing Example

- Things already described with a TD and TDs are linked to each other and describe the composite Thing tree
- Things already discovered
- Data to be linked together: TDs, dynamic data at runtime, static data, tasks



- Composite Thing: series of conveyor belts
  - Tasks: move container from start to end while putting specific workpieces into the container

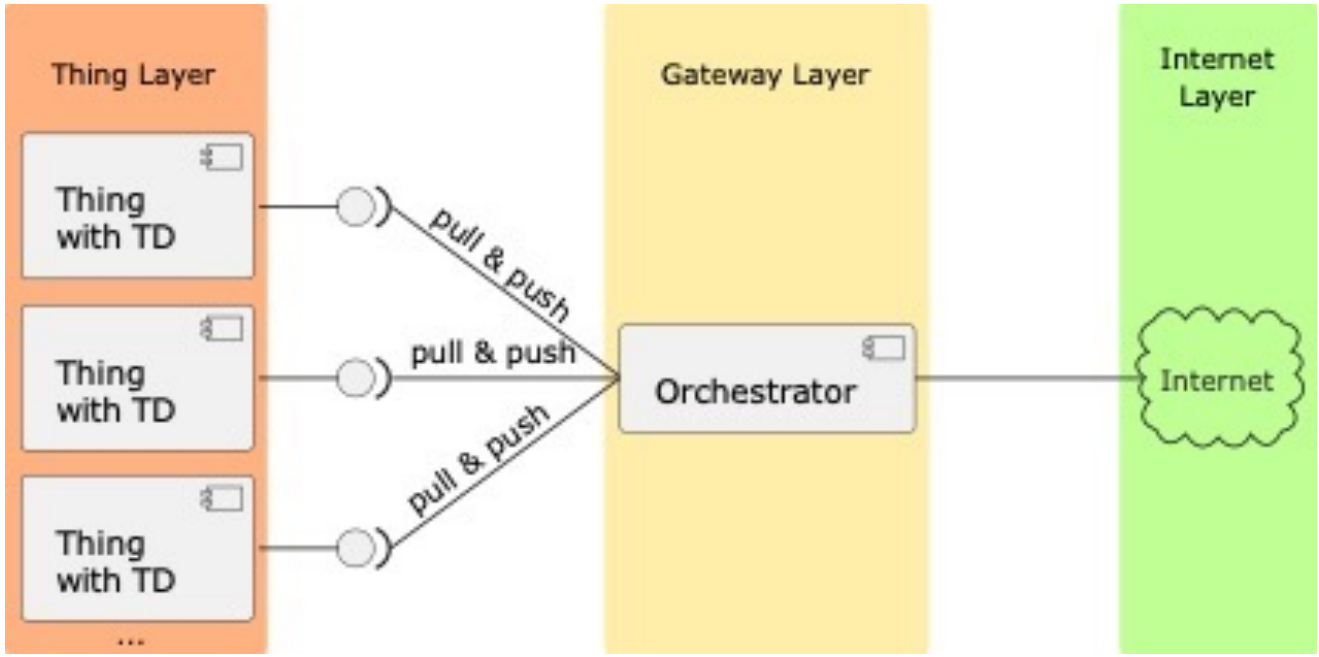


- Subsystems: single conveyor belts

- Standalone Things: Motors that drive the belts, sensors that detect occupancy on a conveyor belt
  - Interaction affordances:
    - Properties: "sensorvalue", "motorstatus"
    - Actions: "runmotor", "stopmotor"

IIoT System under Consideration

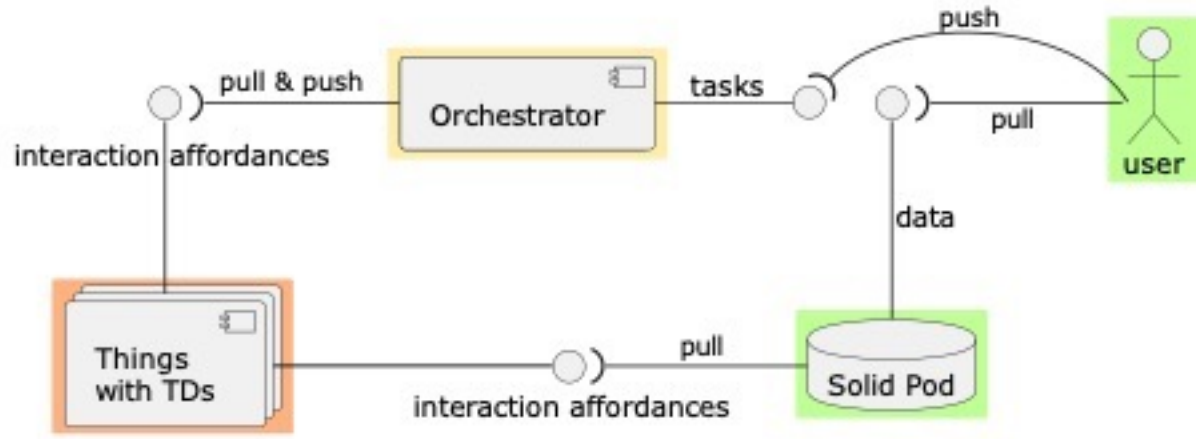
- Thing layer: most constrained
- Internet layer: least constrained
- Industrial computer for control on Gateway layer



Direct Approach

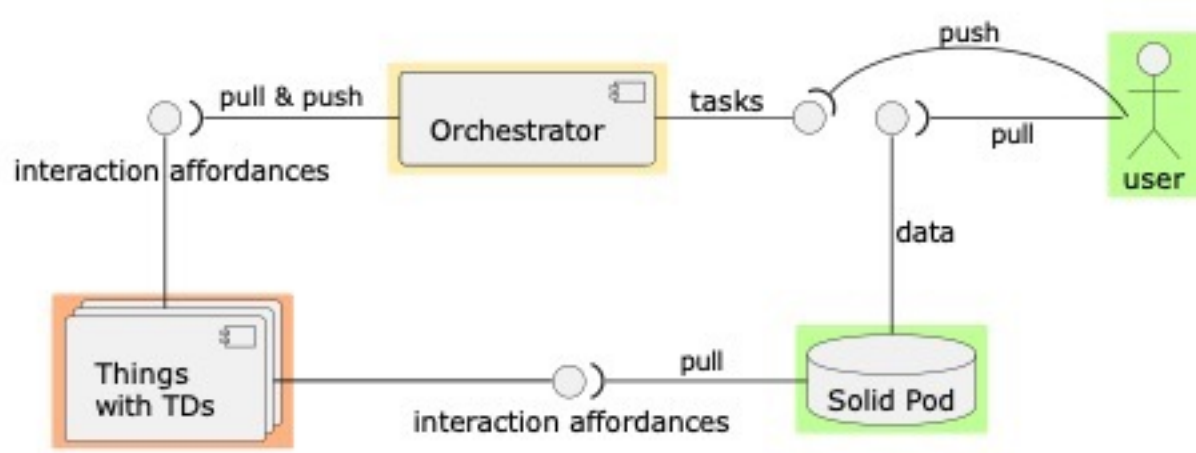
Single Indirection for Interaction and Data

- Newest possible Thing data (Pod forwards queries, transforms to RDF)
- Full Thing data history on Pod
- User interacts with orchestrator to invoke composite Thing
- Pod and user at Internet layer



Direct Approach - Constraints

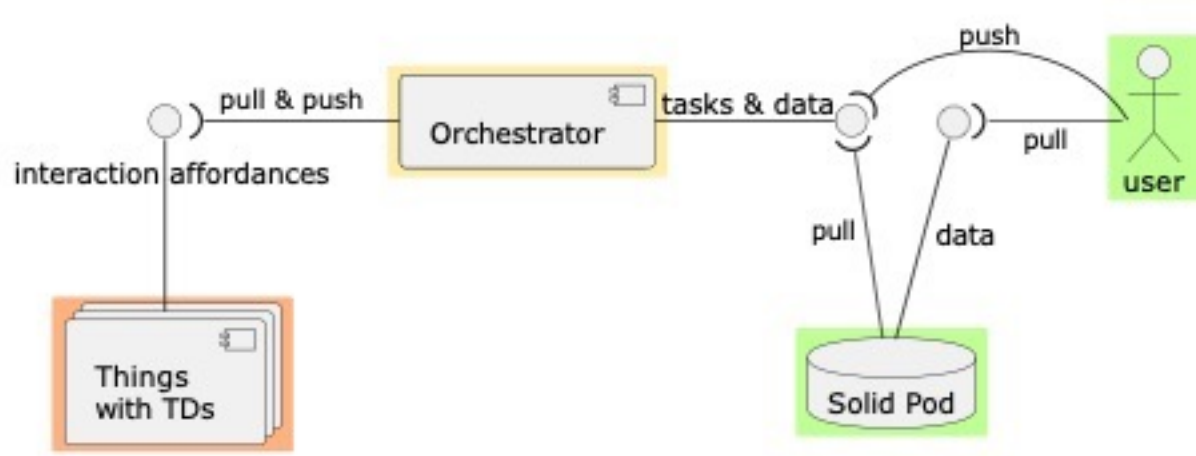
- Things constrained in compute, storage and energy
  - High query frequency from user → high load on Things



Less Direct Approach

Two Indirections for Data, Single Indirection for Interaction

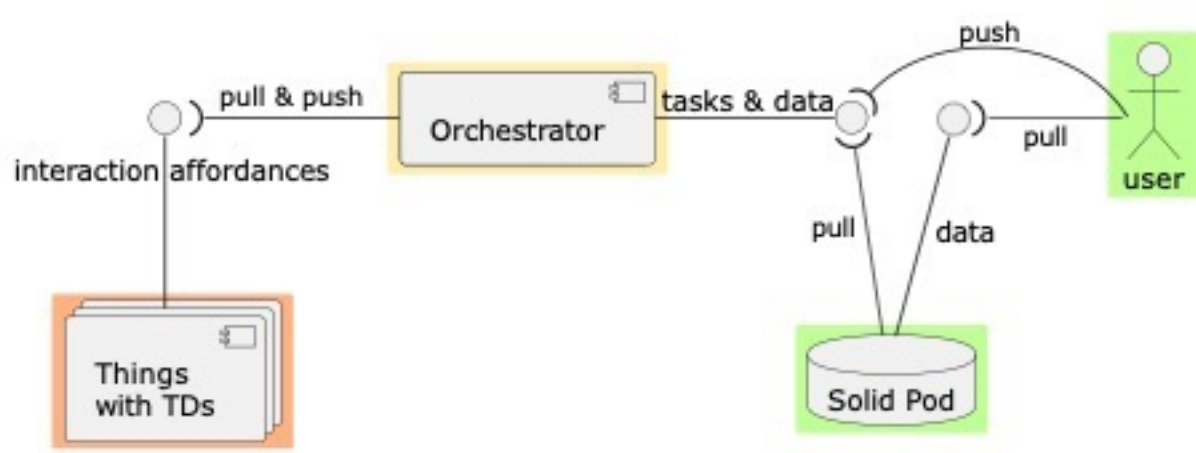
- Things constrained in compute, storage and energy
  - High query frequency from user → high load on Things
- Add orchestrator as indirection, orchestrator already aware of constraints for control logic implementation
- Orchestrator as second query forwarder and task executor



Less Direct Approach

Two Indirections for Data, Single Indirection for Interaction

- Orchestrator constrained in storage
  - Cannot store full task history
- Orchestrator constrained in compute
  - Unconstrained Pod controls communication as client



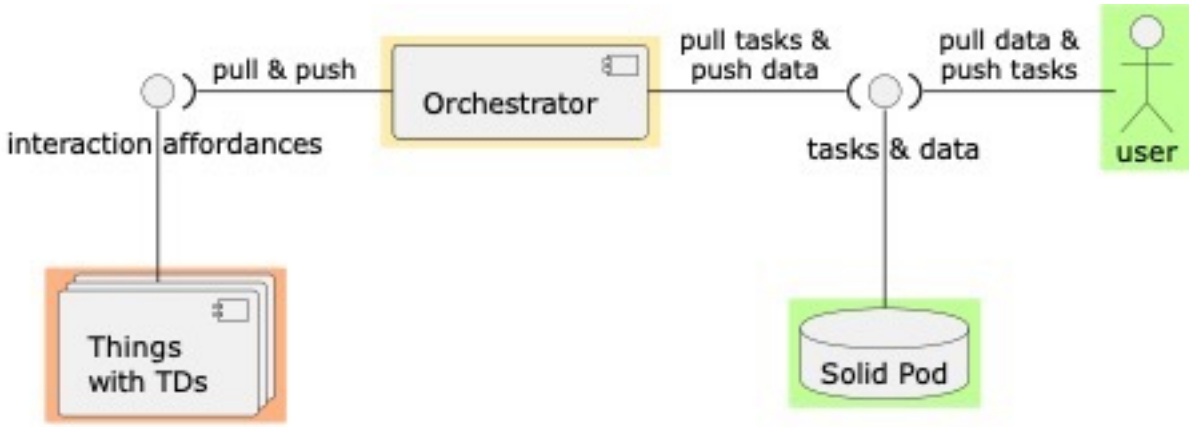
Proposed Architecture

Two Indirections for Data and Interaction

- Orchestrator constrained in storage
  - Cannot store full task history
- Orchestrator constrained in compute
  - Unconstrained Pod controls communication as client

- Tasks stored on Pod, orchestrator pulls tasks
- Orchestrator pushes data fetched during orchestration to Pod (e.g., interval-based)

- Orchestrator is client, Pod and Things are servers



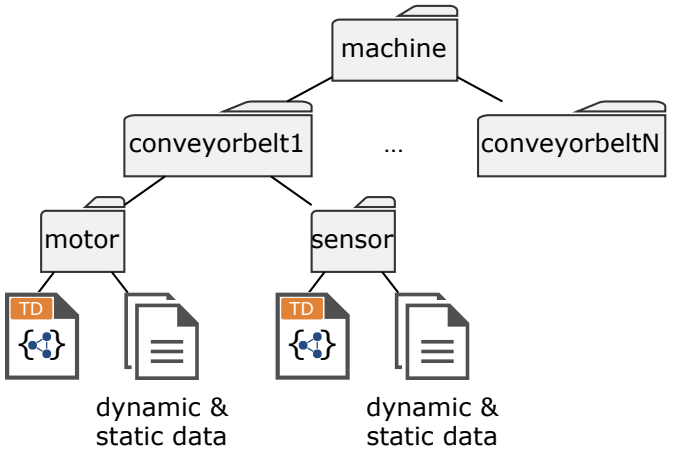
Orchestrator

- Orchestrator is coupled to Things and Pod due to direct communication; Pod and Things are not coupled
  - Adding/removing Things: changes only on the orchestrator, no direct changes on Pod necessary
  - Changes on the Pod (e.g., multiple composite Things on same Pod): changes only on the orchestrator, Things not affected

- Orchestrator aware of all other components → responsible for setting up Pod for interaction and data exchange

Enabling Fine-grained Access Control and Linking Data

- Derive Thing tree from TDs
- Create directory tree based on Thing tree
  - Exposed by Pod as REST API (hierarchy as URI path)
- For data integration, push TDs and static data to respective directories



- Transform runtime data (pulled during task execution) to RDF and link to Things

- Create a task queue directory for users to interact with the composite Thing (Linked Data Notifications [4])
  - e.g., /machine/tasks/
  - Users push tasks as files into directory, composite Thing marks them as done

Performance Limitations

- Two indirections (Pod and orchestrator) when user wants to pull Thing data or push tasks
  - Task queue suited for long-running, uninterruptible tasks
  - Data arrives on Pod based on orchestrator push strategy

Conclusion And Future Work

- Data exchange based on hierarchical composition
- Interaction based on task queue

- Semantic Web for data and device integration
- Off-the-shelf software, custom implementation on orchestrator

- Future work:
  - Apply architecture to more use cases
  - Evolve the architecture
  - Empirical evaluation of the system performance characteristics
  - …

References

[1] M. Kovatsch, R. Matsukura, M. Lagally, T. Kawaguchi, K. Toumura, K. Kajimoto, Web of Things (WoT) Architecture, Recommendation, W3C, 2020. https://www.w3.org/TR/2020/ REC-wot-architecture-20200409/.

[2] S. Käbisch, T. Kamiya, M. McCool, V. Charpenay, M. Kovatsch, Web of Things (WoT) Thing Description, Recommendation, W3C, 2020. https://www.w3.org/TR/2020/ REC-wot-thing-description-20200409/.

[3] S. Speicher, J. Arwe, A. Malhotra, Linked Data Platform 1.0, Recommendation, W3C, 2015. https://www.w3.org/TR/2015/REC-ldp-20150226/.

[4] S. Capadisli, A. Guy, Linked Data Notifications, Recommendation, W3C, 2017. https://www.w3.org/TR/2017/REC-ldn-20170502/.

Thank you for your time

Fraunhofer IIS

Fraunhofer Institute for Integrated Circuits IIS

antrieb 4.0

https://www.antrieb40.org

Grant No. 13IK015B

Finanziert von der Europäischen Union

Gefördert durch:

Bundesministerium für Wirtschaft und Klimaschutz

aufgrund eines Beschlusses des Deutschen Bundestages

Contact

Justus Fries
justus.fries@iis.fraunhofer.de

Center for Applied Research on Supply Chain Services at Fraunhofer Institute for Integrated Circuits IIS
Nordostpark 93
90411 Nürnberg
iis.fraunhofer.de

Fraunhofer IIS

Fraunhofer Institute for Integrated Circuits IIS

Used Software

- Orchestrator implemented using node-wot and NodeJS APIs for Solid (needs read and write access to Pod)
- If Enterprise Solid Server, can generate client credentials for the Pod

Performance Limitations

- Users interact with composite through task queue, that is polled by the orchestrator after finishing a task
  - Suited for long-running, uninterruptible tasks
  - Task execution not directly invokable

- Things only interact with orchestrator to manage constraints while fulfilling tasks
  - Results in delay of data materialization on Pod due to network latency and push strategy on orchestrator
  - Enables hiding direct Thing interfaces from users if needed (remove forms from TDs on Pod)