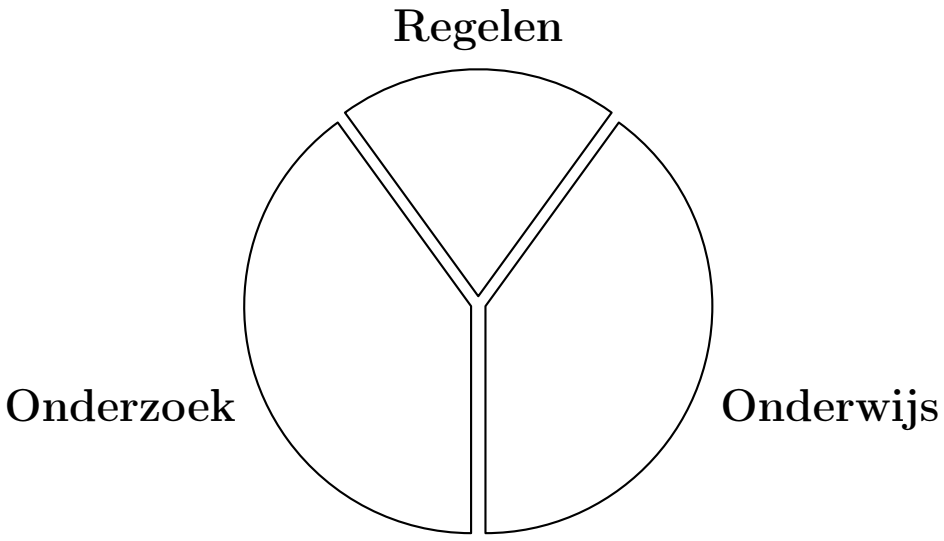
Universitair Docent

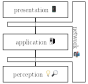

Regelen
Onderzoek
Onderwijs

Tierless Internet of Things with Task-Oriented Programming

Mart Lubbers (mart@cs.ru.nl)
Radboud University Nijmegen

Radboud University

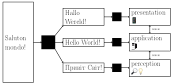
Challenges in the Internet of Things (IoT)



Typical IoT software architecture

- Estimated 20 billion ($20 \cdot 10^9$) devices
- Heterogeneous architectures
- Semantic friction
- Interoperation problems

Tierless programming

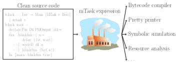


Typical tierless IoT architecture.

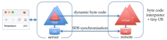
All layers and interoperation is generated from single: source, language, paradigm, and type system

Powered by task-oriented programming

The mTask language is a DSL implemented in Clean

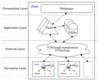


The mTask system integrates seamlessly with the iTask system.



University of Glasgow case study

		tiered	tierless
single-board computer		PRS	CRS
micro controller		PWS	CWS



- real-world external example
- four-way comparison
- functional requirements
  - measure environment
  - centralised database
  - web interface to data
  - management and monitoring of devices

Tierless (CWS, CRS) leads to 70%-90% reduction in SLOC compared to tiered (CRS, PRS).

Software metrics

Number of paradigms and languages.

	tiered		tierless	
category	PWS	PRS	CWS	CRS
edge	μPython	Python	mTask	iTask
server	Python, JSON Redis, MongoDB PHP, HTML		iTask	
total	7	6	2	1
para	imp. dec.	imp. dec.	decl.	decl.

Number of files and lines of code.

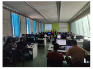
	tiered		tierless	
category	PWS	PRS	CWS	CRS
sensor node	52	57	11	11
sensor interface	178	183	9	4
communication	76	76	35	30
webinterface	56	56	28	28
database	106	106	12	12
management	94	98	5	4
total	562	576	100	89
files	35	38	3	5

less paradigms/languages
↓
fewer interoperation problems

less code
↓
easier maintenance

Conclusions

- Tierless programming simplifies the creation and maintenance of IoT applications.
- Using DSLs, existing tierless languages can be extened to work on IoT edge devices.
- Proven itself in the UoG case study[2].
- Used by students in three summer schools: 3COWS 2019, SusTrainable 2022 and 2023.



SusTrainable 2023, Rijeka, Croatia

For more information

Reference 1

Reference 2

Get mTask here!

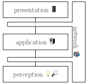
References

1. M. Lubbers, Orchestrating the Internet of Things with Task-Oriented Programming. in Radboud Dissertation Series, no. DIS-002. Nijmegen: Radboud University Press, 2023. doi: 10.54195/9789493296114.
2. M. Lubbers, P. Koopman, A. Ramsingh, J. Singer, and P. Trinder, 'Could Tierless Languages Reduce IoT Development Grief?'. in ACM Trans. Internet Things, vol. 4, no. 1, Feb. 2023. doi: 10.1145/3572901.

Tierless Internet of Things with Task-Oriented Programming
Mart Lubbers (mart@cs.ru.nl)
Radboud University Nijmegen

Radboud University

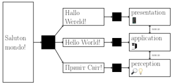
Challenges in the Internet of Things (IoT)



Typical IoT software architecture

- Estimated 20 billion $(20 \cdot 10^9)$ devices
- Heterogeneous architectures
- Semantic friction
- Interoperation problems

Tierless programming

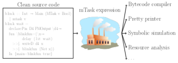


Typical tierless IoT architecture.

All layers and interoperation is generated from single: source, language, paradigm, and type system

Powered by task-oriented programming

The mTask language is a DSL implemented in Clean

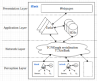


The mTask system integrates seamlessly with the iTask system.

University of Glasgow case study

		tiered	tierless
single-board computer		PRS	CRS
micro controller		PWS	CWS

- real-world external example
- four-way comparison
- functional requirements
  - measure environment
  - centralised database
  - web interface to data
  - management and monitoring of devices



Tierless (CWS, CRS) leads to 70%-90% reduction in SLOC compared to tiered (CRS, PRS).

Software metrics

Number of paradigms and languages.

	tiered		tierless	
category	PWS	PRS	CWS	CRS
edge	μPython	Python	mTask	iTask
server	Python, JSON Redis, MongoDB PHP, HTML		iTask	
total	7	6	2	1
para	imp. dec.	imp. dec.	decl.	decl.

Number of files and lines of code.

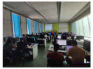
	tiered		tierless	
category	PWS	PRS	CWS	CRS
sensor node	52	57	11	11
sensor interface	178	183	9	4
communication	76	76	35	30
webinterface	56	56	28	28
database	106	106	12	12
management	94	98	5	4
total	562	576	100	89
files	35	38	3	5

less paradigms/languages
↓
fewer interoperation problems

less code
↓
easier maintenance

Conclusions

- Tierless programming simplifies the creation and maintenance of IoT applications.
- Using DSLs, existing tierless languages can be extened to work on IoT edge devices.
- Proven itself in the UoG case study[2].
- Used by students in three summer schools: 3COWS 2019, SusTrainable 2022 and 2023.


SusTrainable 2023, Rijeka, Croatia

For more information

Reference 1 Reference 2 Get mTask here!

References

1. M. Lubbers, Orchestrating the Internet of Things with Task-Oriented Programming. in Radboud Dissertation Series, no. DIS-002. Nijmegen: Radboud University Press, 2023. doi: 10.54195/9789493296114.
2. M. Lubbers, P. Koopman, A. Ramsingh, J. Singer, and P. Trinder, 'Could Tierless Languages Reduce IoT Development Grief?'. in ACM Trans. Internet Things, vol. 4, no. 1, Feb. 2023. doi: 10.1145/3572901.

Could Tierless Languages Reduce IoT Development Grief?

MART LUBBERS, Radboud University, Netherlands
PIETER KOOPMAN, Radboud University, Netherlands
ADRIAN RAMSINGH, University of Glasgow, United Kingdom
JEREMY SINGER, University of Glasgow, United Kingdom
PHIL TRINDER, University of Glasgow, United Kingdom

Internet of Things (IoT) software is notoriously complex, conventionally comprising multiple tiers. The developer must use multiple programming languages and ensure that the components interoperate correctly. A novel alternative is to use a single *tierless* language with a compiler that generates the code for each component and ensures their correct interoperation.

We report a systematic comparative evaluation of two tierless language technologies for IoT stacks: one for resource-rich sensor nodes (Clean with iTask), and one for resource-constrained sensor nodes (Clean with iTask and mTask). The evaluation is based on four implementations of a typical smart campus application: two tierless and two Python-based tiered. (1) We show that tierless languages have the potential to significantly reduce the development effort for IoT systems, requiring 70% less code than the tiered implementations. Careful analysis attributes this code reduction to reduced interoperation (e.g. two embedded domain-specific languages (DSLs) and one paradigm versus seven languages and two paradigms), automatically generated distributed communication, and powerful IoT programming abstractions. (2) We show that tierless languages have the potential to significantly improve the reliability of IoT systems, describing how Clean iTask/mTask maintains type safety, provides higher order failure management, and simplifies maintainability. (3) We report the first comparison of a tierless IoT codebase for resource-rich sensor nodes with one for resource-constrained sensor nodes. The comparison shows that they have similar code size (within 7%), and functional structure. (4) We present the first comparison of two tierless IoT languages, one for resource-rich sensor nodes, and the other for resource-constrained sensor nodes.

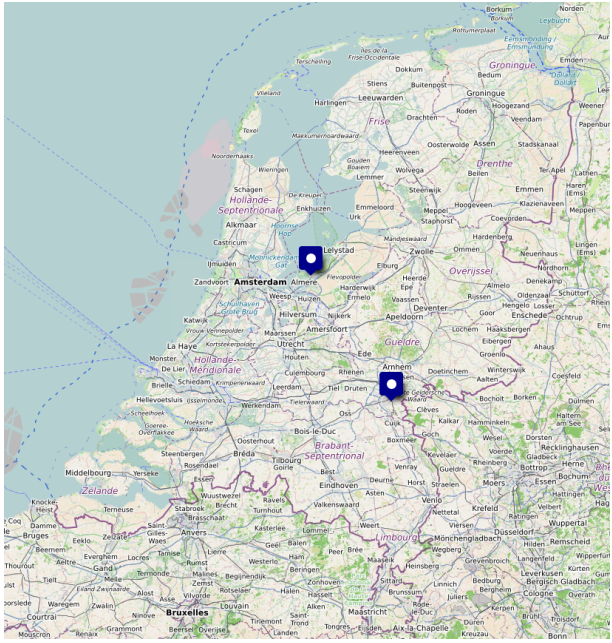
CCS Concepts: • **Computer systems organization** → *Sensor networks*; **Embedded software**; • **Software and its engineering** → *Domain specific languages*.

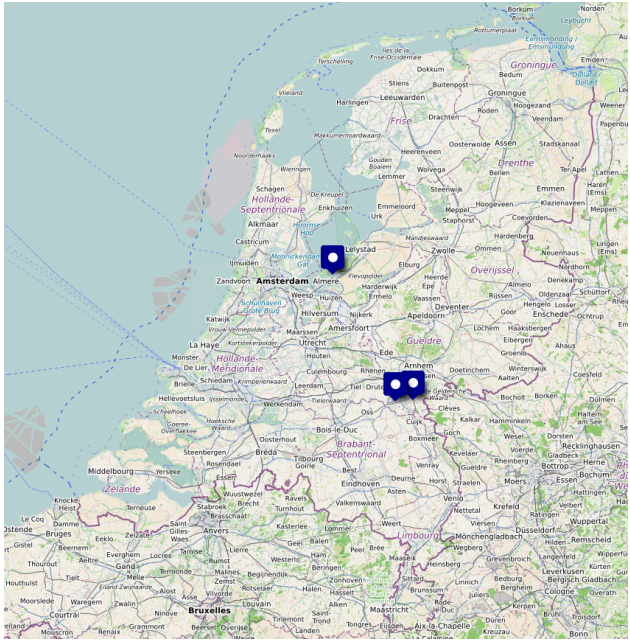
Additional Key Words and Phrases: Tierless languages, IoT stacks

ACM Reference Format:
Mart Lubbers, Pieter Koopman, Adrian Ramsingh, Jeremy Singer, and Phil Trinder. 2021. Could Tierless Languages Reduce IoT Development Grief?. *ACM Trans. Internet Things* 37, 4, Article 111 (August 2021), 35 pages. https://doi.org/10.1145/1122445.1122456

Authors' addresses: Mart Lubbers, Radboud University, P.O. Box 9010, Nijmegen, Netherlands, firstname@cs.ru.nl; Pieter Koopman, Radboud University, P.O. Box 9010, Nijmegen, Netherlands, firstname@cs.ru.nl; Adrian Ramsingh, University of Glasgow, 17 Lilybank Gardens, Glasgow, United Kingdom, firstname.lastname@glasgow.ac.uk; Jeremy Singer, University of Glasgow, 17 Lilybank Gardens, Glasgow, United Kingdom, firstname.lastname@glasgow.ac.uk; Phil Trinder, University of Glasgow, 17 Lilybank Gardens, Glasgow, United Kingdom, firstname.lastname@glasgow.ac.uk.

2577-6207/2021/8-ART111 $15.00
https://doi.org/10.1145/1122445.1122456


Amsterdam
Bruxelles
Hollande-Septentrionale
Hollande-Méridionale
Frise
Groningue
Drenthe
Overijssel
Gueldre
Brabant-Septentrional
Zélande
Limbourg
Lelystad
Arnhem


Amsterdam
Bruxelles
Lelystad
Arnhem
Hollande-Septentrionale
Hollande-Méridionale
Frise
Groningue
Drenthe
Overijssel
Gueldre
Utrecht
Brabant-Septentrional
Zélande
Limbourg


Radboud University