Institute for Enterprise Systems
Empowering Digital Transformation

UNIVERSITY OF MANNHEIM

Emergent Software Service Platform and ist Application in a Smart Mobility Setting

Christoph Knieke, Eric Nyakam, Andreas Rausch, Christian Schindler, Christian Bartelt, Nils Wilken, Nikolaus Ziebura

Nils Wilken
wilken@es.uni-mannheim.de
Institute for Enterprise Systems (InES), University of Mannheim, Germany

IARIA

Who I Am

- Short CV
  - 2013 – 2016: Bachelor of Science in Business Informatics at the University of Mannheim
  - 2016 – 2018: Master of Science in Business Informatics at the University of Mannheim
  - Since 2018: Researcher at the Institute for Enterprise Systems, University of Mannheim

- Research Interests
  - Goal Recognition
  - Plan Recognition
  - Artificial Intelligence

Institute for Enterprise Systems
Empowering Digital Transformation

UNIVERSITY OF MANNHEIM

Motivation

IARIA

Motivation

- One major characteristic of IoT environments is a high level of dynamism:
  - Available services/devices change constantly at runtime
  - Context information in the environment might change frequently
  - User needs and requirements might change in response to changes of the environment

- This is a major challenge for software systems, as they have to be able to change their behavior to adapt to dynamic changes in their environments

- Currently, these challenges are already addressed by Dynamic Adaptive Systems and Self-adaptive Systems

Emergent Software Service Platform: Required Capabilities

1. Automatic elicitation of user requirements at runtime.

2. Automatic composition of a software service, which meets the user requirements.

3. Automatic execution of a composed software service at runtime.

4. Providing the execution result to the platform user(s).

Institute for Enterprise Systems
Empowering Digital Transformation

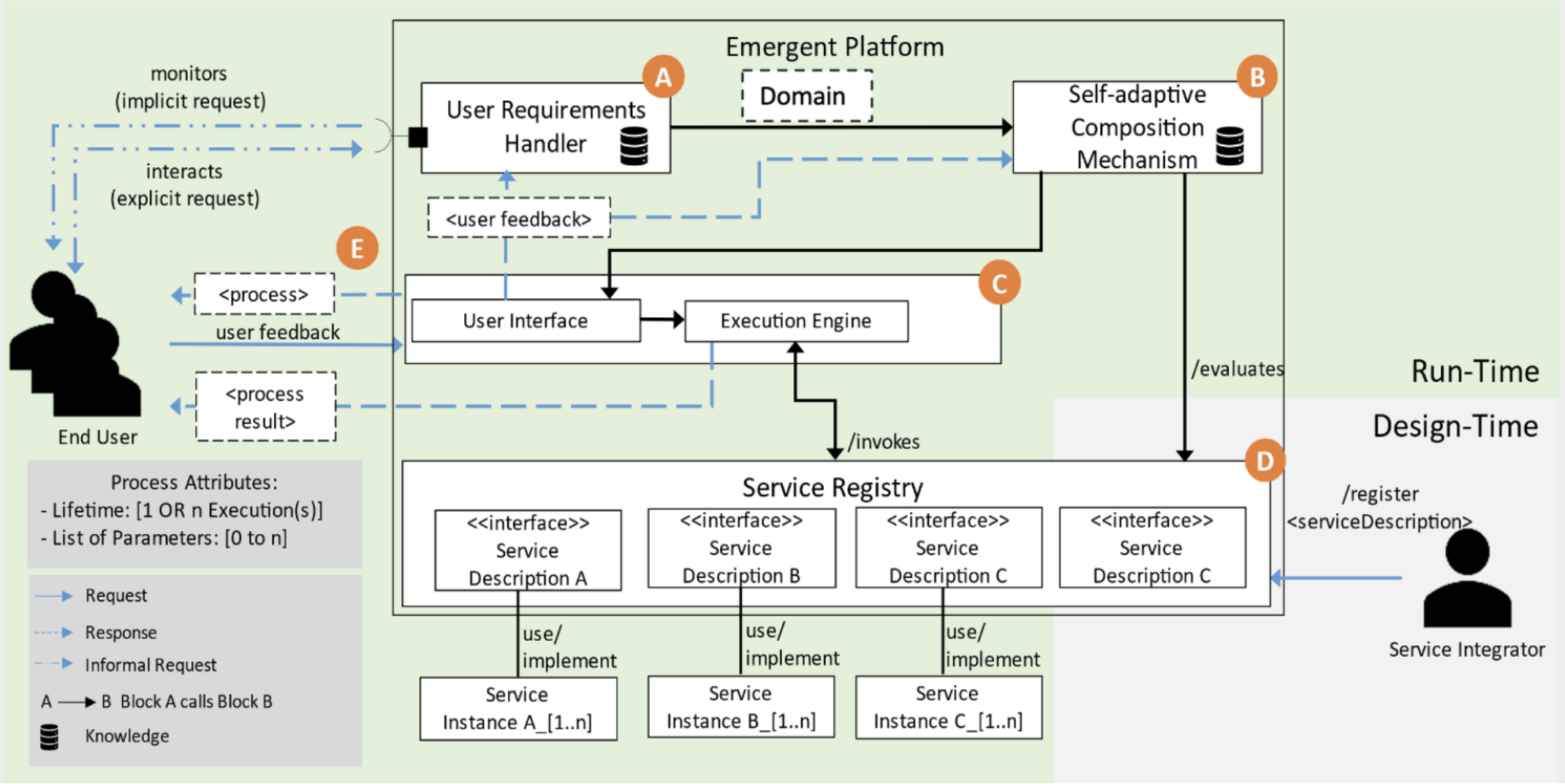
UNIVERSITY OF MANNHEIM

Emergent Software Service Platforms: Architecture

IARIA

Emergent Software Platform: Definition

„A software platform is called emergent if it automatically and dynamically composes available components in response to a trigger event. The resulting behaviour of the platform is not predefined at design time and not anticipated by the individual components.“


Emergent Platform
monitors
(implicit request)
interacts
(explicit request)
A
User Requirements Handler
Domain
B
Self-adaptive Composition Mechanism
<user feedback>
E
<process>
user feedback
<process result>
End User
C
User Interface
Execution Engine
/evaluates
/invokes
Run-Time
Design-Time
D
Service Registry
<<interface>> Service Description A
<<interface>> Service Description B
<<interface>> Service Description C
<<interface>> Service Description C
/register
<serviceDescription>
Service Integrator
use/ implement
use/ implement
use/ implement
Service Instance A_[1..n]
Service Instance B_[1..n]
Service Instance C_[1..n]
Process Attributes:
- Lifetime: [1 OR n Execution(s)]
- List of Parameters: [0 to n]
Request
Response
Informal Request
A B Block A calls Block B
Knowledge

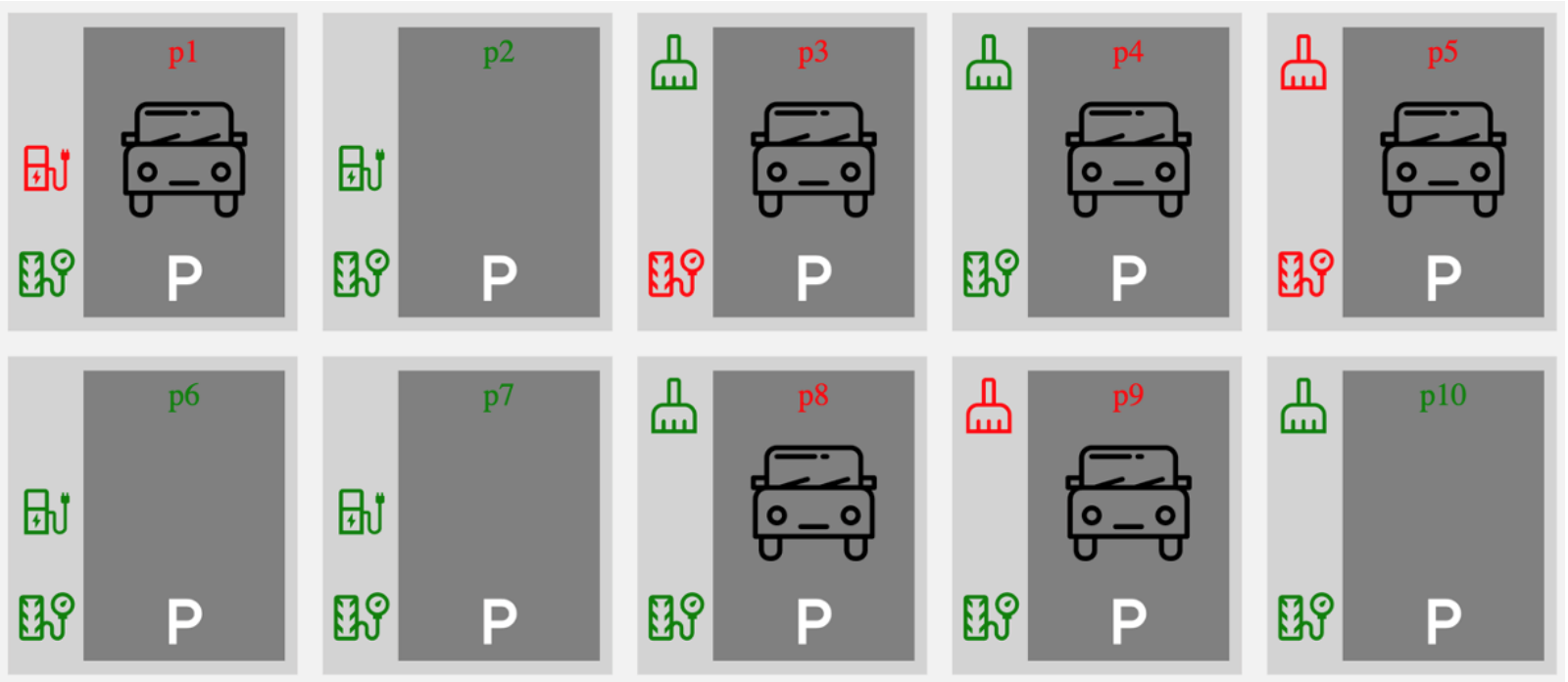
Institute for Enterprise Systems
Empowering Digital Transformation

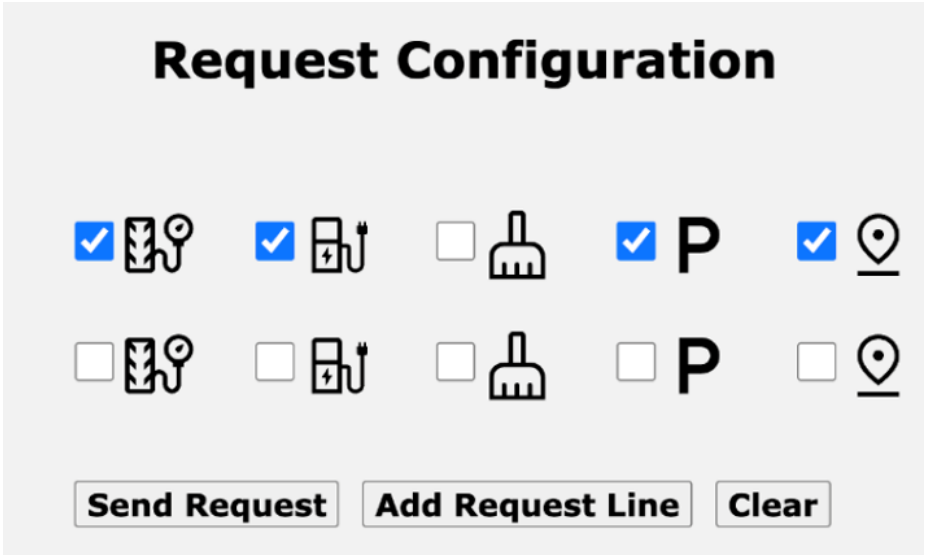
UNIVERSITY OF MANNHEIM

Emergent Software Service Platform: Application to a Smart Mobility Setting

IARIA

Application to a Smart Mobility Setting: Parking Lot UI





Application to a Smart Mobility Setting: Formalized Internal Platform Communication


```
{"environment": [
 {"value":"", "type":"parkingid", "name":"p1"},
 {"value":"", "type":"operatorid", "name":"b1"},
 {"value":"", "type":"reservationnr", "name":"r1"},
 {"value":"", "type":"maxparkingtime", "name":"m1"},
 {"value":"", "type":"bookedservice", "name":"g1"}],
"init": [],
"goal": "(and (tirepressurecheck r1)
(bookeparking p1 r1 m1)
(navigation p1))"}
```

Application to a Smart Mobility Setting: Formalized Internal Platform Communication


```
{"composition": [
 {"name":"get_parking-e-available",
    "params":["p1","b1"]},
 {"name":"post_book-parking-e", "params":
    ["p1", "r1", "b1", "m1"]},
 {"name":"book-tirepressurecheck",
    "params":["p1", "m1", "r1"]},
 {"name":"get_parking-navigation-parkingid",
    "params":["p1"]}],
"environment": [
 {"value":"", "type":"parkingid", "name":"p1"},
 {"value":"", "type":"operatorid", "name":"b1"},
 {"value":"", "type":"reservationnr", "name":"r1"},
 {"value":"", "type":"maxparkingtime", "name":"m1"},
 {"value":"", "type":"bookedservice", "name":"g1"}]}
```

Conclusion and Future Work

Conclusion and Future Work

- A prototypical implementation of the presented architecture is able to automatically elicit user requirements from an explicit request
- Automatic composition and execution of a software service is possible as an answer to recognized user requirements
- One limitation is that the considered use case is rather small.

Questions

Thank you for your attention!

Do you have any questions?

For follow up questions you can contact me at any time (wilken@es.uni-mannheim.de)