



## Internet on Wheels

Workshop Testnet

10 juli 2017

Ruben Wildeboer & Rix Groenboom

## Agenda

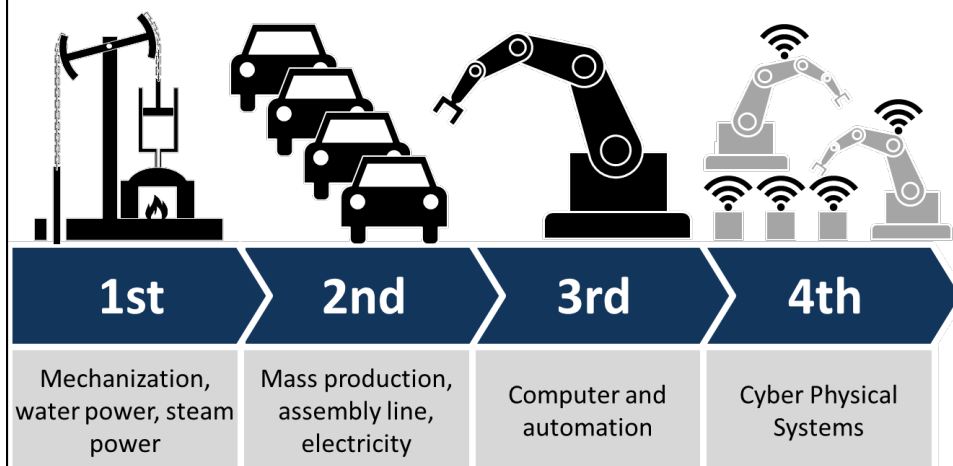
- Introductie
  - Uitdelen USB ivm installeren software
- Internet of Things
- DASH
- Pauze:
  - Software activeren
- Aan de slag met Dash

## Line-up van vandaag

- Parasoft
  - Ruben / Rix
  - Chantal / Karina
- Dash
- Amazing Maze
- SOAtest / Virtualize
  - Software installeren (gedurende introductie)
  - Licentie-sleutel (in de pauze)



## 4<sup>de</sup> industriële revolutie



## 4<sup>de</sup> industriële revolutie

- Cyber Physical Systems
  - Systems of Systems
- “Computer got out of his cage”
  - Flying and ridings drones
- “Internet of Things”
  - Belangrijk onderdeel van deze revolutie

## 4<sup>de</sup> industriële revolutie

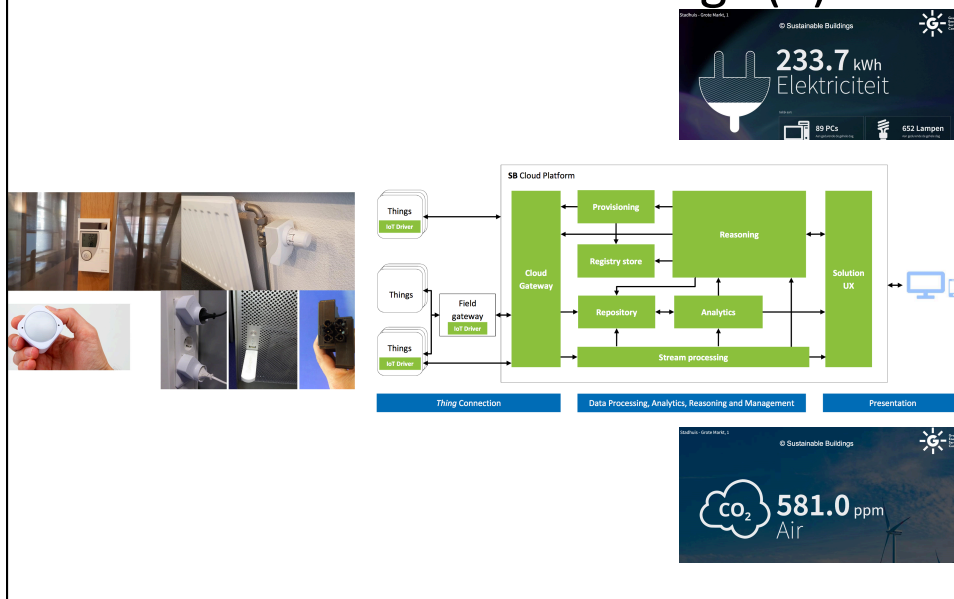
- Cyber Physical Systems
  - Systems of Systems
- “Computer got out of his cage”
  - Flying and ridings drones
- “Internet of Things”
  - Belangrijk onderdeel van deze revolutie
  - Toepassingen in de “fysieke wereld”

## IoT: Connected systems

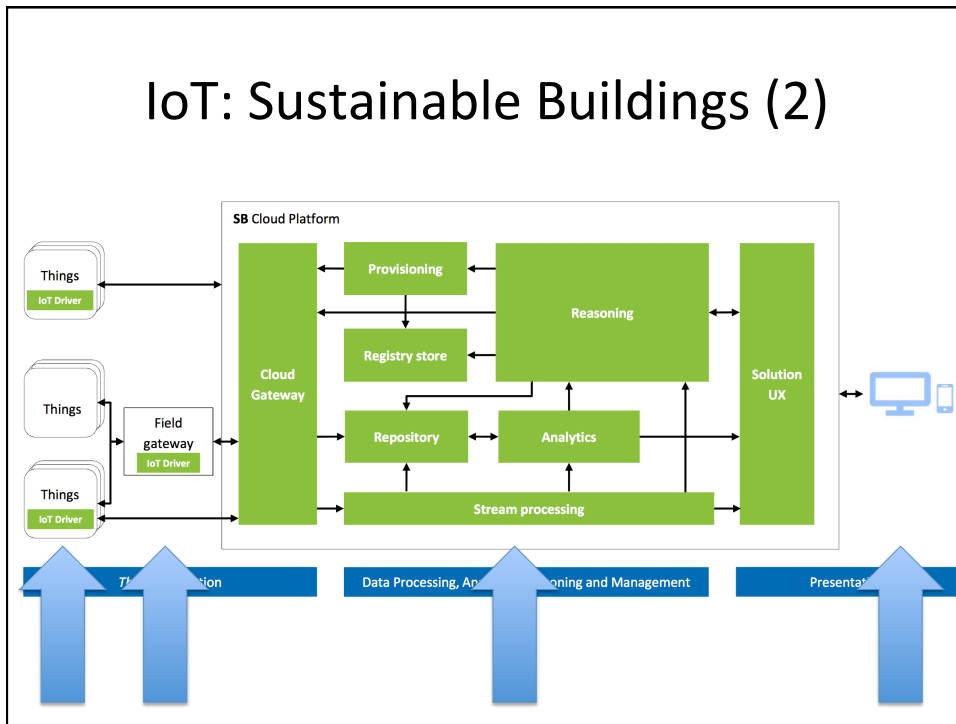
- Zr.Ms. De Ruyter (F104)
- Radar systems (SMART-L)



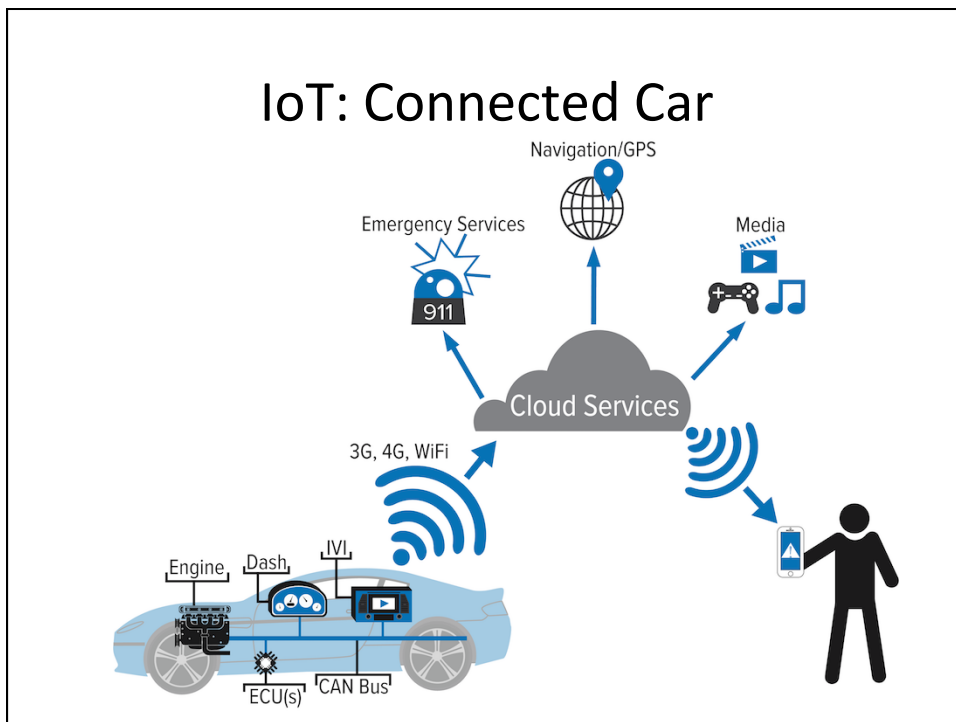
## IoT: Sustainable Buildings (1)



## IoT: Sustainable Buildings (2)



## IoT: Connected Car



## IoT: Dash, onze connected car



## DASH

- Educational Robot
  - Programmeren met Apps
- Sensoren:
  - Geluid, afstand, beweging
- Actuatoren:
  - Geluid, rijden, kleuren
- Communicatie:
  - Bluetooth Low Energy (BLE) protocol

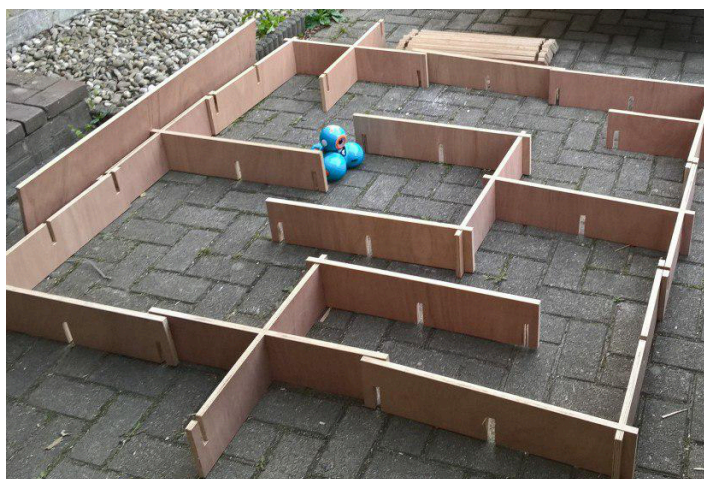


## DOEL: WIE-KENT-KWIS

- Caviabak uit de Wie-Kent-Kwis (vanaf min 1):  
<https://www.youtube.com/watch?v=i2NIhn1gKX>



## DOEL: Amazing Maze



## Test setup (1)

- Parasoft tooling communicatie



## Test setup (2)

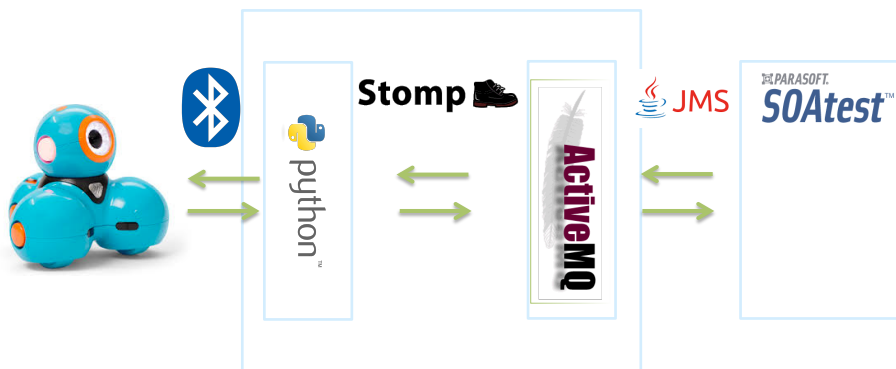
- Parasoft tooling communicatie





## Test setup (3)

- Parasoft tooling communicatie



## Connectivity

- DASH: Spreekt Bluetooth Low Energy (BLE)
- Python (Ruben.py): BLE & STOMP
  - Polling van Dash
  - Omzetten van berichten in commando's
- ActiveMQ: STOMP & JMS
  - STOMP voor interne communicatie
  - JMS voor externe communicatie



## Twee soorten aansturing

1. Pro-actief / vanuit een model:
  - a. Direct commando's
  - b. Interactief (reageren op omgeving)
2. Reactief, vanuit sensor data (a la Cavia)
  - a. Bij botsing een keuze maken



## Twee soorten ontwerpen

1. Programma direct op Robot uitproberen (geen testomgeving)
2. Programma testen met gesimuleerde data (gebruik van virtualize voor testen)





## AAN DE SLAG

### Nu de praktijk (1)

SOAtest:

- Test tool voor oa JMS communicatie
- Diverse commando's versturen, eventueel vanuit een tabel (spreadsheet)
- Call-back tool voor het ontvangen van (asynchrone) berichten, en daar op te reageren
- Extra logica via Python



## Nu de praktijk (2)

### Virtualize:

- Simulatie tool voor o.a. JMS communicatie
- Diverse commando's versturen, op basis van binnenkomende events
- Proxies om live-verkeer te volgen / recording

### Algemeen:

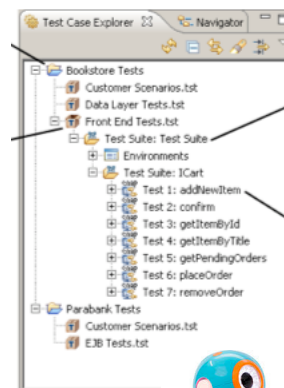
- Queue Browser om JMS-queues te volgen



## Nu de praktijk (3)

### Eclipse Workspace

- Eclipse Project
  - SOAtest test file: FOO.tst
    - Environments
    - Datasources
    - Properties
    - Test suites
      - SOAPClient
      - Dbtool
      - Callback tool



## Connectivity (1)

### ActiveMQ (op laptop van Ruben)

- Instelling:
  - tcp://192.168.1.100:61616
  - org.apache.activemq.jndi.ActiveMQInitialContextFactory
  - ConnectionFactory
- Queues:
  - DashRequest
  - DashResponse



## Connectivity (2)

Request Transport Success Criteria

Transport: JMS

Connection Settings

- Queue/Topic
- Messaging Model
- Message Exchange Pattern
- Message Expiration
- Message Type
- Request Message Properties
- Response Message Correlation

**JNDI Initial Context**

Settings	Properties
Provider URL:	Fixed tcp://\$(HOST):61616
Initial Context:	Fixed org.apache.activemq.jndi.ActiveMQI
Connection Factory:	Fixed ConnectionFactory

**Queue or Topic Connection Authentication**

Perform authentication

## Connectivity (3)

Instructions Test 1: drive:500

Name: drive:500 Data Source: testdata

Schema URL

Request Transport Success Criteria

Transport: JMS

Connection Settings

- Queue/Topic
- Messaging Model
- Message Exchange Pattern
- Message Expiration
- Message Type
- Request Message Properties
- Response Message Correlation

JMSDestination

Fixed DashRequest

JMSReplyTo

From Fixed JMS\_ResponseQ

## Connectivity (4)

Window Help

- Minimize
- Zoom
- Toggle Full Screen ^⌘F
- New Window
- Editor
- Appearance
- Show View
- Perspective
- Navigation
- Bring All to Front

drive:500 - Parasoft SOAtest & Virtualize

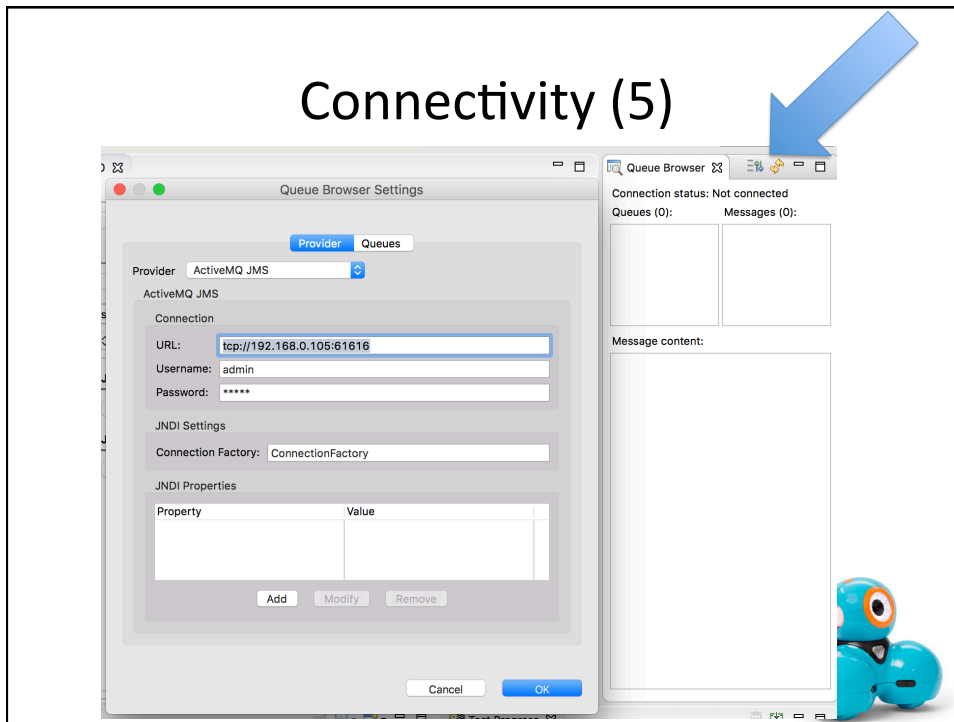
Quick Access

Data Source

Data Source: testdata

- Console ⌘Q C
- Data Repositories
- Navigator
- Quality Tasks
- Queue Browser
- Scanned Resources Listing
- SOAtest Server
- Task List ⌘Q K
- Test Case Explorer
- Test Progress
- Other... ⌘Q Q

## Connectivity (5)



## Commando's (1)

- move:dist:speed
  - dist in mm
  - speed in mm/sec
- turn:degr:speed
  - degr in graden
  - speed in mm/sec
- stop
- ear\_color:red,green,white,blue,orange



## Commando's (2)

instructies Test 1: drive:500

**General**

Name: instructies Type: Table

**Rows**

All Range From: 1 To: 11

**Table**

First row specifies column names

	A	B	C	D	E
	command	arg1	arg2	wait	
1	drive	50		5000	
2	stop				
3	turn	-90	100	1000	
4	drive	50		4000	
5	turn	90	100	1000	
6	drive	50		4000	
7	turn	90	100	1000	
8	drive	50		4000	
9	turn	-90	100	1000	
10	drive	50		5000	
11	stop				
12					



## Commando's (3)

instructies Test 1: drive:500 Test 1: do-it

**Name** Name: do-it **Data Source** Data Source: instructies

**Schema URL**

Constrain to Schema Refresh

**Request** **Transport** **Success Criteria**

Input Mode: Literal

Text  File  Data Source

MIME Type: text/xml

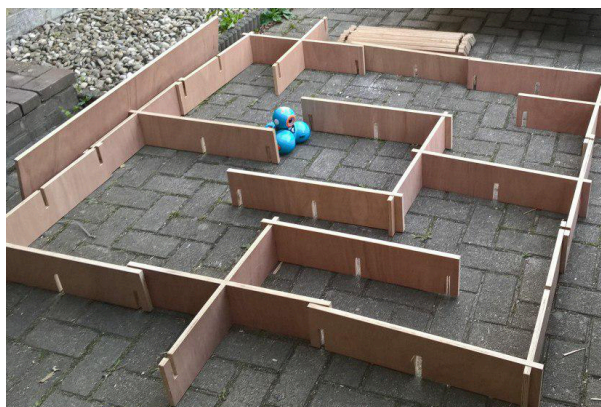
```
1 ${command}:${arg1}:${arg2}
```





## DOEL: Amazing Maze

- 2:30 minuten voor Dash om uitgang te halen  
– <https://www.youtube.com/watch?v=ECfeMI92-o8>



AAN DE SLAG

## Sensor data gebruiken

Interactief reageren door sensor-data te gebruiken:

- JMS Queue: DashResponse
- Berichten als:
  - `<prox_right>20</prox_right>`
  - `<prox_left>70</prox_left>`
  - `<moving>1</moving>`
  - `<roll>100</roll>`
  - `<pitch>100</pitch>`
  - `<acceleration>1100</acceleration>`



## Sensor data gebruiken (1)

- SOAtest heeft “Call Back tool” om queues te lezen:

Test 1: read sensor

**Name**

Name:

**Tool Settings**

Timeout (sec):

**Protocol**

Parlay
  Parlay X
  SCP
  WS-Addressing
  Custom
  No correlation
  JMS
  TIBCO
  Web

**Connection Settings**

Queue/Topic:

Messaging Model

Message Expiration

Message Type

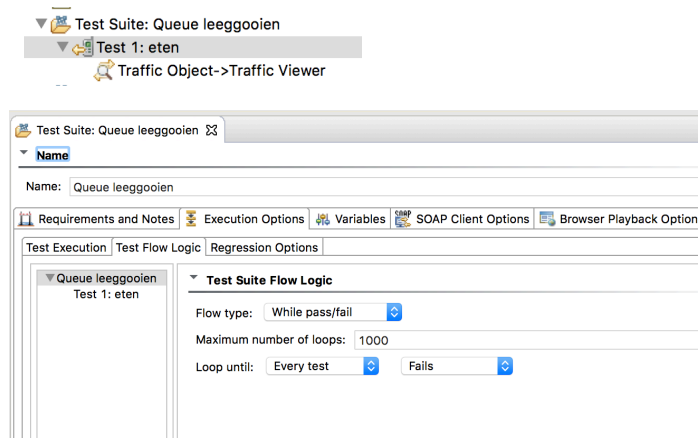
Incoming Message Correlation

JMS Address:



## Initialiseren

Queues leeghalen (bij start):



The screenshot shows a test suite configuration window for 'Queue leeggooien'. The 'Name' field is set to 'Queue leeggooien'. Below the name, there are tabs for 'Requirements and Notes', 'Execution Options', 'Variables', 'SOAP Client Options', and 'Browser Playback Option'. The 'Test Execution' tab is active, showing 'Test Suite Flow Logic' with the following settings:

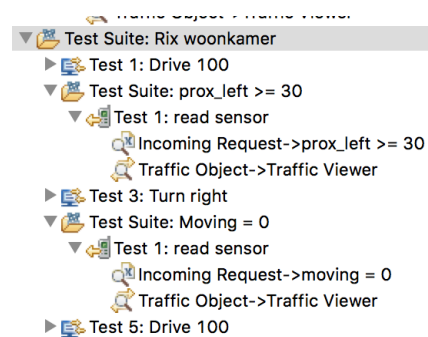
- Flow type: While pass/fail
- Maximum number of loops: 1000
- Loop until: Every test

A small blue robot icon is visible in the bottom right corner of the screenshot.

## Algoritme

Standaard loopje:

- Drive
- Check afstand tot muur
- Draai
- Wacht tot je stil staat



The screenshot shows a test suite configuration window for 'Rix woonkamer'. The test suite contains the following tests:

- Test 1: Drive 100
- Test Suite: prox\_left >= 30
  - Test 1: read sensor
    - Incoming Request->prox\_left >= 30
    - Traffic Object->Traffic Viewer
- Test 3: Turn right
- Test Suite: Moving = 0
  - Test 1: read sensor
    - Incoming Request->moving = 0
    - Traffic Object->Traffic Viewer
- Test 5: Drive 100

A small blue robot icon is visible in the bottom right corner of the screenshot.

## Hoe nu sensor checken

Polling sensor:

- While loop tot status is bereikt
- Lees sensor data
  - Gebruik callback tool
- Check tot bepaalde status optreed
  - Gebruik XML assertor
  - Op Xpath van bericht, bijvoorbeeld:
    - `/moving/text() = 0`
    - `/prox_left/text() >= 30`



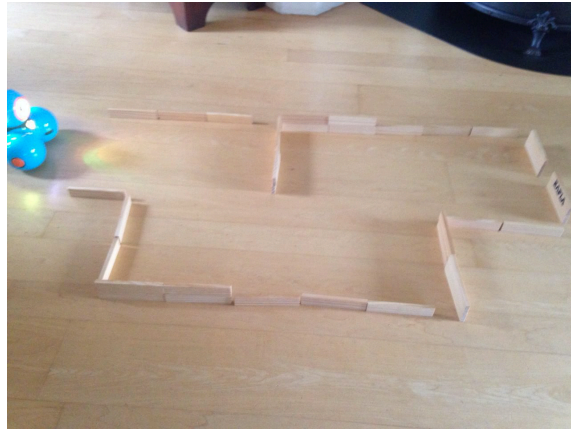
## Hoe nu sensor checken

XML assertor:

The screenshot displays a software interface for configuring XML assertions. The main window is titled "moving = 0" and shows a "Numeric Assertion" configuration. The "Name" field is set to "moving = 0". The "Tool Settings" section includes tabs for "Summary", "Configuration", and "Expected XML". The "Configuration" tab is active, showing "Name: Numeric Assertion" and "Numeric Assertion Configuration" with "Element must be" set to "==" and "expected value" set to "Fixed" with a value of "0". A blue arrow points from the "Configuration" tab to an "Edit Element" dialog box. The "Edit Element" dialog has two radio buttons: "Select element graphically from tree" (selected) and "Edit XPath manually". The "XPath" field contains "/moving/text()". A small blue robot icon is in the bottom right corner.

# Programma

- ▼ Test Suite: Test Suite
  - ▢ Environments
  - ▶ Test Suite: Queue leeggoien
  - ▼ Test Suite: Rix woonkamer
    - ▶ Test 1: Drive 100
    - ▶ Test Suite: prox\_left >= 30
    - ▶ Test 3: Turn right
    - ▶ Test Suite: Moving = 0
    - ▶ Test 5: Drive 100
    - ▶ Test Suite: prox\_left >= 30
    - ▶ Test 7: Turn left
    - ▶ Test 8: ear red
    - ▶ Test Suite: Moving = 0
    - ▶ Test 10: Drive 100
    - ▶ Test Suite: prox\_left >= 30
    - ▶ Test 12: Turn left
    - ▶ Test Suite: Moving = 0
    - ▶ Test 14: Drive 4
    - ▶ Test Suite: prox\_left >= 30
    - ▶ Test 16: Turn right 2
    - ▶ Test Suite: Moving 4
    - ▶ Test 18: Drive 100
    - ▶ Test Suite: prox\_left >= 30
    - ▶ Test 20: stop



NEXT STEPS

## Vervolg stappen

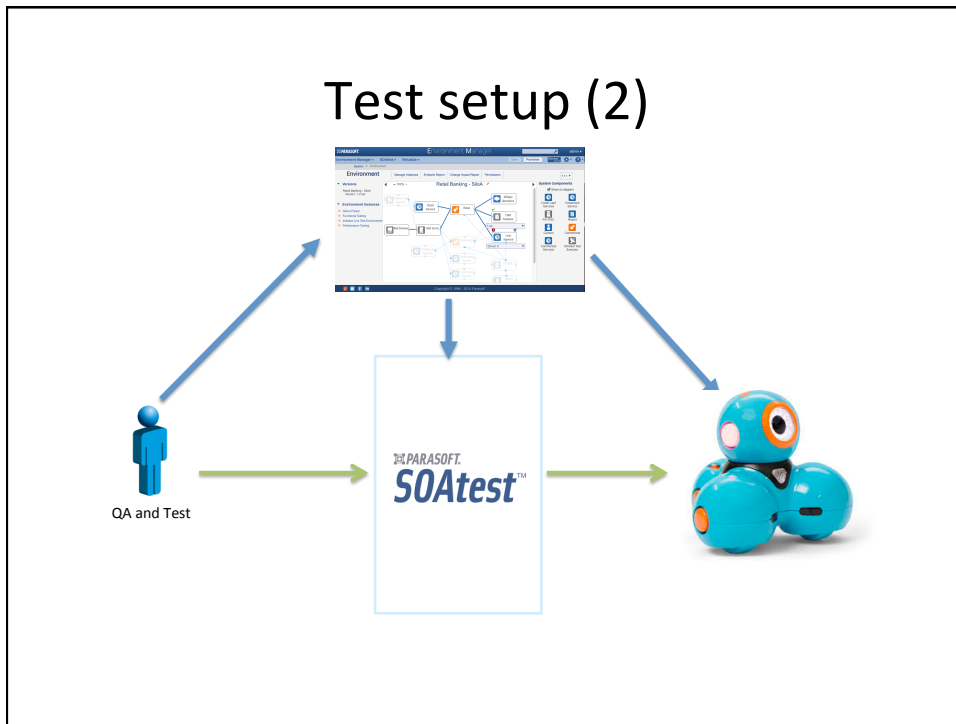
- Combinatie van 2 aanpakken:
  - “model” (XLS) en sensor data
- Simuleren van sensor data:
  - Recording van gegevens
  - Tijd-registratie
- Simuleren van complete “maze”



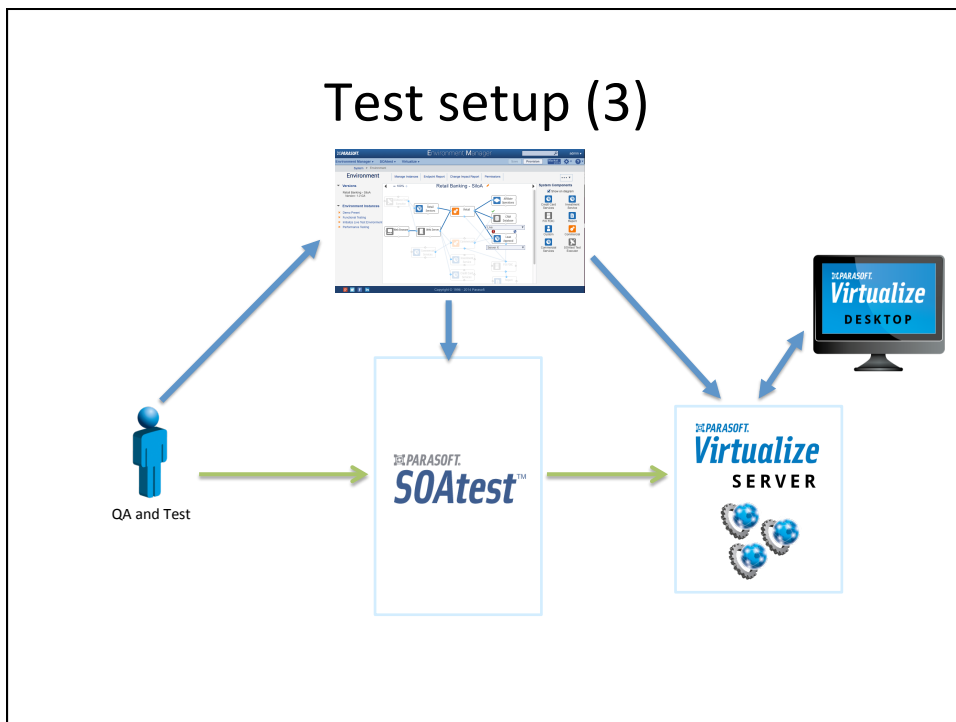
## Test setup (1)



## Test setup (2)



## Test setup (3)





## APPENDIX