Core Activities

- Digital Twins & Process Simulation
- Algorithm & Software Development
- Machine Diagnostics

Contact details

Office: Meensesteenweg 385-389, B-8501 Kortrijk (Bissegem)
Tel: +32 (0) 495 216 227
Email: info@vintecc.com
CONTENT

- Vintecc
- Virtual commissioning
- Virtual commissioning of an AGV using sensor simulation
DIGITAL TWINS & SIMULATION
• Process Simulation
• Virtual commissioning
• Automated testing

ALGORITHMS & SOFTWARE
• Process Control
• Sensor Technologies
• Code generation

MACHINE DIAGNOSTICS
• Component / System Diagnostics
• Data Analytics
• IoT
Our Competencies

TRENDS

Faster Time-to-market
- ROI (+More resources for innovation)
- Early mover advantage

System Safety / Quality
- IEC 61131-3 compliant code
- Requirements tracing&verification

Complexity / Performance
- Closer interaction of multiple eng disciplines
- More automation = more sensors & software

Market Trends

Our Clients

Machine Simulation
- Black/white box modeling
- Design verification (MIL/SIL/HIL)

Machine Software
- Control System development
- Auto Code Generation

Smart Sensors
- Vision
- Soft Sensors

Agile Development
- Rapid Prototyping
- Auto Code Generation
- Hardware Independent
CONTENT

- Vintecc
- Virtual commissioning
- Virtual commissioning of an AGV using sensor simulation
VIRTUAL COMMISSIONING VS CONVENTIONAL SOFTWARE DEVELOPMENT

MECHANICS

ENGINEER & DESIGN

PROCURE & SUPPLY

MANUFACTURE & ASSEMBLE

SOFTWARE & ELECTRICS

INSTALL ELECTRICS

DEVELOP SW

TEST

CAD/CAM

3D CAD

3D CAD

BOM

CAM

VIRTUAL MACHINE

REAL MACHINE

VIRTUAL COMMISSIONING

CONVENTIONAL SOFTWARE DEVELOPMENT
VIRTUAL COMMISSIONING VS CONVENTIONAL SOFTWARE DEVELOPMENT

**Engineer & Design**
- 3D CAD

**Procure & Supply**
- BOM
- CAM

**Manufacture & Assemble**
- Virtual Machine

**Test**
- Install Electrics
- Rapid Prototyping

**Simulation Environment**
- Digital Twins
- MIL / SIL / HIL

**Commission**
VIRTUAL COMMISSIONING - WHY?

<table>
<thead>
<tr>
<th>Conventional software development</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Software

Real Machine

Actuators

Sensors

CPU
**VIRTUAL COMMISSIONING - WHY?**

<table>
<thead>
<tr>
<th>Conventional software development</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Physical Machine</strong></td>
</tr>
<tr>
<td>• Availability</td>
</tr>
<tr>
<td>• Expensive and limited testing opportunity</td>
</tr>
<tr>
<td><strong>Maintainability/complexity</strong></td>
</tr>
<tr>
<td>• Every change is a risk of breaking something</td>
</tr>
<tr>
<td>• Struggle to deliver in time keeping up with competition</td>
</tr>
</tbody>
</table>

| **Sequential Development**        |
| • Switching costs increase exponentially |
| • Slow Iterative development cycles |

Software   

- [CPU]  
- [Sensors]  
- [Actuators]  

Real Machine
# Virtual Commissioning - Why?

<table>
<thead>
<tr>
<th>Conventional software development</th>
<th>Virtual commissioning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Physical Machine</strong></td>
<td></td>
</tr>
<tr>
<td>• Availability</td>
<td></td>
</tr>
<tr>
<td>• Expensive and limited testing opportunity</td>
<td></td>
</tr>
<tr>
<td><strong>Maintainability/complexity</strong></td>
<td></td>
</tr>
<tr>
<td>• Every change is a risk of breaking something</td>
<td></td>
</tr>
<tr>
<td>• Struggle to deliver in time keeping up with competition</td>
<td></td>
</tr>
<tr>
<td><strong>Sequential Development</strong></td>
<td></td>
</tr>
<tr>
<td>• Switching costs increase exponentially</td>
<td></td>
</tr>
<tr>
<td>• Slow Iterative development cycles</td>
<td></td>
</tr>
</tbody>
</table>

![Diagram showing the comparison between conventional software development and virtual commissioning](image)

- **Software**
- **Real Machine**
  - Actuators
  - Sensors
• Vintecc

• Virtual commissioning

• Virtual commissioning of an AGV using sensor simulation
STOW PALLET SHUTTLE

• The Company
  • Averys group, 750 employees
  • Worldwide service organisation
  • Pallet racking and shelving systems

• The project
  • Custom controller board and legacy code base preventing new feature development
  • Conversion to full model-based software
    • Series product based on beckhoff target
    • Hardware independence due to model based workflow
    • Simulation (sensors, drives,...) allows to add new and complex features ... fast
<table>
<thead>
<tr>
<th>WHY?</th>
<th>Conventional software development</th>
<th>Virtual commissioning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Physical Machine</strong></td>
<td>• Availability (concept vs variant)</td>
<td><strong>Digital Twin</strong></td>
</tr>
<tr>
<td></td>
<td>• Expensive and limited testing opportunity</td>
<td></td>
</tr>
<tr>
<td><strong>Maintainability/complexity</strong></td>
<td>• Every change is a risk of breaking something</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Struggle to deliver in time keeping up with competition</td>
<td></td>
</tr>
<tr>
<td><strong>Sequential Development</strong></td>
<td>• Switching costs increase exponentially</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Slow Iterative development cycles</td>
<td></td>
</tr>
</tbody>
</table>

[Diagram showing software and virtual machine connections with sensors and actuators]
**WHY?**

- Import from CAD
- Sensor simulation

---

**Virtual commissioning**

**Digital Twin**
- Variant is a parameter
- Test (hazardous, expensive) situations at endless repeatability
<table>
<thead>
<tr>
<th>Why？</th>
<th>Conventional software development</th>
<th>Virtual commissioning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Physical Machine</strong></td>
<td></td>
<td><strong>Digital Twin</strong></td>
</tr>
<tr>
<td>• Availability (concept vs variant)</td>
<td>• Variant is a parameter</td>
<td></td>
</tr>
<tr>
<td>• Expensive and limited testing opportunity</td>
<td>• Test (hazardous, expensive) situations at endless repeatability</td>
<td></td>
</tr>
<tr>
<td><strong>Maintainability/complexity</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Every change is a risk of breaking something</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Struggle to deliver in time keeping up with competition</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Sequential Development</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Switching costs increase exponentially</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Slow Iterative development cycles</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Diagram](diagram.png)
<table>
<thead>
<tr>
<th>WHY?</th>
<th>Conventional software development</th>
<th>Virtual commissioning</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Physical Machine</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Availability (concept vs variant)</td>
<td>• Variant is a parameter</td>
</tr>
<tr>
<td></td>
<td>• Expensive and limited testing opportunity</td>
<td>• Test (hazardous, expensive) situations at endless repeatability</td>
</tr>
<tr>
<td></td>
<td><strong>Maintainability/complexity</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Every change is a risk of breaking something</td>
<td>• Graphical state machines</td>
</tr>
<tr>
<td></td>
<td>• Struggle to deliver in time keeping up with competition</td>
<td>• Software as a competitive advantage</td>
</tr>
<tr>
<td></td>
<td><strong>Sequential Development</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Switching costs increase exponentially</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Slow Iterative development cycles</td>
<td></td>
</tr>
</tbody>
</table>

Software models

Virtual Machine

Actuators

Sensors

Graphical programming
Virtual commissioning

Graphical programming

- Graphical state machines
- Software as a competitive advantage

Why?

Application

Utilities

Drivers

Physical IO

Simulink generated components

Legacy

IOLINK  Canopen  DIO  RS232  SAFETY

SAFETY

DS402

Graphical state machines

Software as a competitive advantage
## WHY?

### Conventional software development

<table>
<thead>
<tr>
<th>Physical Machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Availability (concept vs variant)</td>
</tr>
<tr>
<td>• Expensive and limited testing opportunity</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Maintainability/complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Every change is a risk of breaking something</td>
</tr>
<tr>
<td>• Struggle to deliver in time keeping up with competition</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sequential Development</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Switching costs increase exponentially</td>
</tr>
<tr>
<td>• Slow Iterative development cycles</td>
</tr>
</tbody>
</table>

### Virtual commissioning

<table>
<thead>
<tr>
<th>Digital Twin</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Variant is a parameter</td>
</tr>
<tr>
<td>• Test (hazardous, expensive) situations at endless repeatability</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Graphical programming</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Graphical state machines</td>
</tr>
<tr>
<td>• Software as a competitive advantage</td>
</tr>
</tbody>
</table>

### Software models

VIRTUAL MACHINE

- Sensors
- Actuators
- Graphical programming
- Software models
## Why?

<table>
<thead>
<tr>
<th>Conventional software development</th>
<th>Virtual commissioning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Physical Machine</strong></td>
<td><strong>Digital Twin</strong></td>
</tr>
<tr>
<td>• Availability (concept vs variant)</td>
<td>• Variant is a parameter</td>
</tr>
<tr>
<td>• Expensive and <strong>limited</strong> testing opportunity</td>
<td>• Test (hazardous, expensive) situations at endless repeatability</td>
</tr>
<tr>
<td><strong>Maintainability/complexity</strong></td>
<td><strong>Graphical programming</strong></td>
</tr>
<tr>
<td>• Every change is a risk of breaking something</td>
<td>• Graphical state machines</td>
</tr>
<tr>
<td>• Struggle to deliver in time keeping up with competition</td>
<td>• Software as a competitive advantage</td>
</tr>
<tr>
<td><strong>Sequential Development</strong></td>
<td><strong>Co-development</strong></td>
</tr>
<tr>
<td>• Switching costs increase exponentially</td>
<td>• Co-simulation of mechanics, sensors &amp; software from day 1</td>
</tr>
<tr>
<td>• Slow Iterative development cycles</td>
<td>• Make right choices from the start</td>
</tr>
</tbody>
</table>

**Software models**

**Virtual Machine**

Sensors

Actuators
## Conventional software development

**Physical Machine**
- Availability (concept vs variant)
- Expensive and limited testing opportunity

**Maintainability/complexity**
- Every change is a risk of breaking something
- Struggle to deliver in time keeping up with competition

**Sequential Development**
- Switching costs increase exponentially
- Slow Iterative development cycles

## Virtual commissioning

**Digital Twin**
- Variant is a parameter
- Test (hazardous, expensive) situations at endless repeatability

**Graphical programming**
- Graphical state machines
- Software as a competitive advantage

**Co-development**
- Co-simulation of mechanics, sensors & software from day 1
- Make right choices from the start

---

**Software**

- How to get back to real machine?
- What can the models be used for?

**Real Machine**

- Actuators
- Sensors
VIRTUAL COMMISSIONING – WORKFLOW – REUSE OF MODELS

1. Simulation

2. Real time testing

3. Automated testing (HIL / PIL /MIL)

Statemachines
Sensor simulation
3D/2D/1D
Static vs dynamic
Scopes
Breakpoints

External mode
Twincat measurement
Breakpoints
ADS interface/soft realtime

Database of scenarios
SENSOR SIMULATION – 3D

- Sensor simulation not limited to 1D
CONCLUSION – VIRTUAL COMMISSIONING – WHY?

1. Decrease development time
   • Fast development of complex machine/features
     • Start when no hardware is available (or variant not available)
     • "One-click" from Concept Validation to Rapid Prototyping

2. Manage complexity
   • Graphical programming
     • Easy to understand & explain
     • Manage Statemachines
     • Self-Documenting (generate HTML)
   • Auto-Code generation
     • Avoid manual coding mistakes

3. Increase maintainability
   • Hardware independent software
     • From Model to PLC / C / C++ code
   • Reproduce in-field scenarios
     • Special scenario’s (safety!)
     • Test particular situation that occurred at customer
   • Training of customers & service engineers
… and many other use cases
Thanks for your attention

AND TELL US ABOUT YOUR PROJECT!