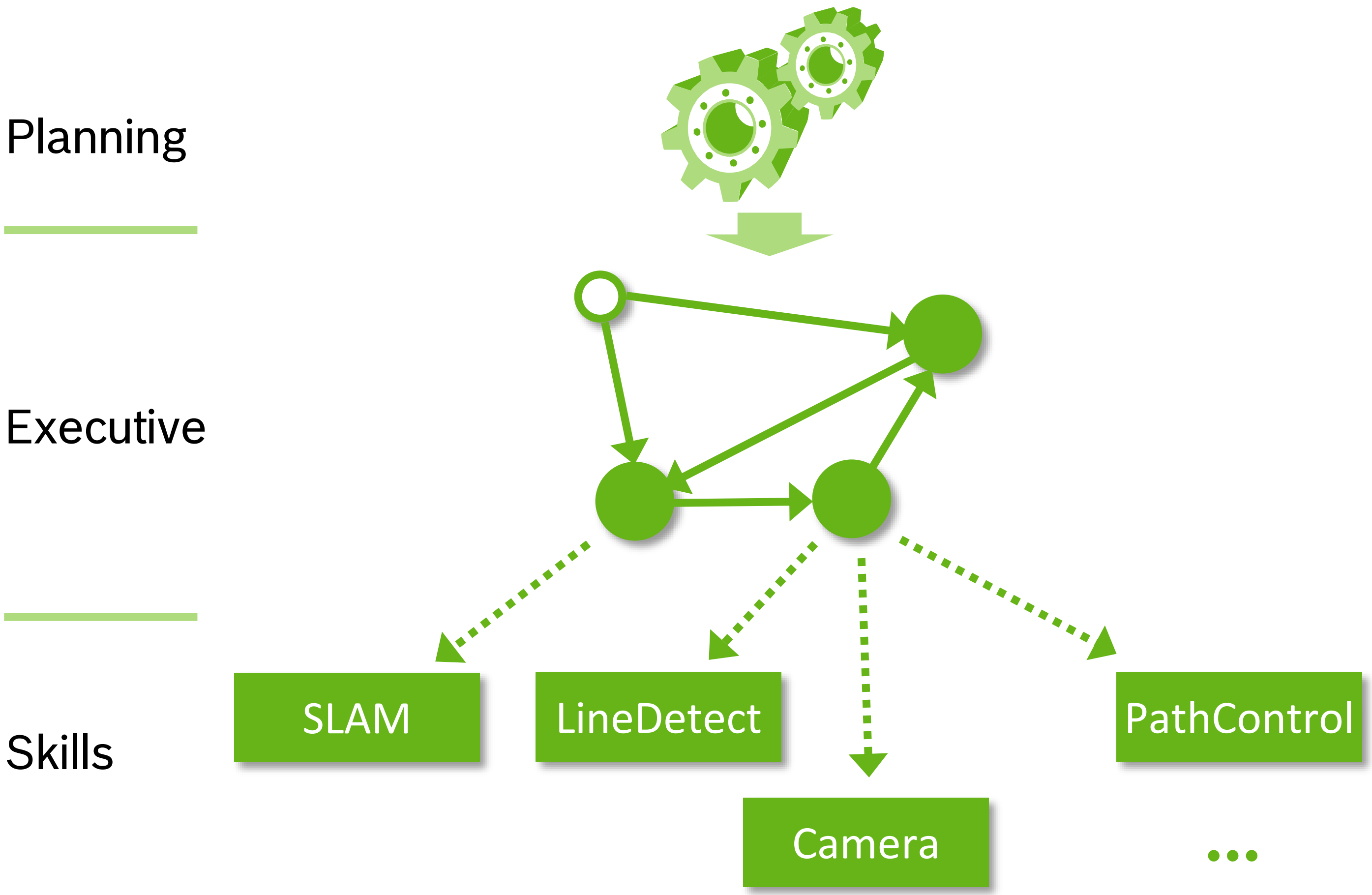
# vTSL – A Formally Verifiable DSL for Specifying Robot Tasks

Christian Heinzemann and Ralph Lange

Robert Bosch GmbH
Corporate Sector Research and Advance Engineering
Renningen, Germany
[firstname.lastname]@de.bosch.com

# vTSL – A Formally Verifiable DSL for Specifying Robot Tasks
## Motivation

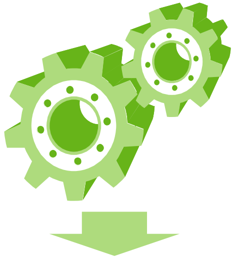




CR/AEE4-Heinzemann, CR/AEE1-Lange | 2018-10-04

# vTSL – A Formally Verifiable DSL for Specifying Robot Tasks
## Motivation

Planning



```
...
camera.config(ACTIVE)
lineDetect.config(ACTIVE)
pathControl.driveLinear(0.3)
...
if (FOUND_LINE) then
   ...
end
```

Executive

Skills

SLAM

LineDetect

Camera

PathControl

...

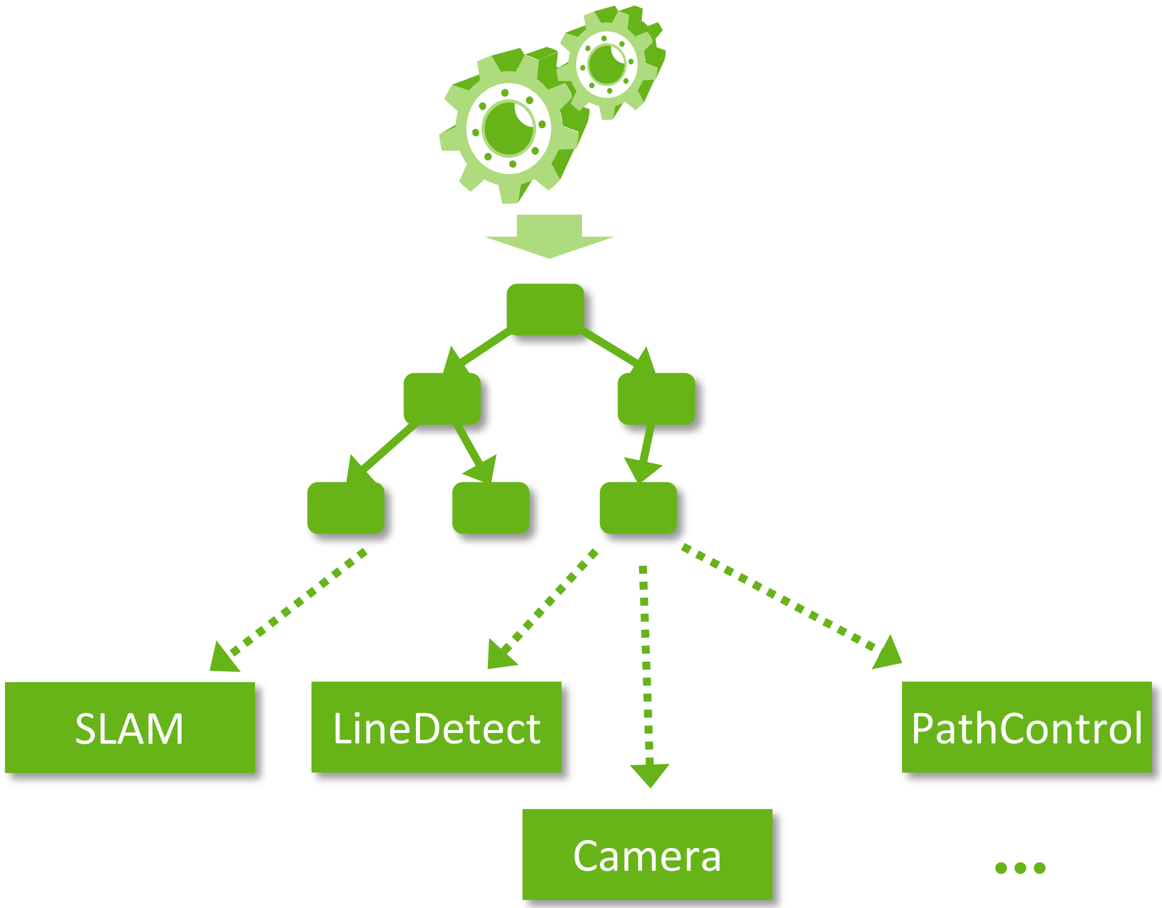

CR/AEE4-Heinzemann, CR/AEE1-Lange | 2018-10-04

# vTSL – A Formally Verifiable DSL for Specifying Robot Tasks
## Motivation

Planning

Executive

Skills





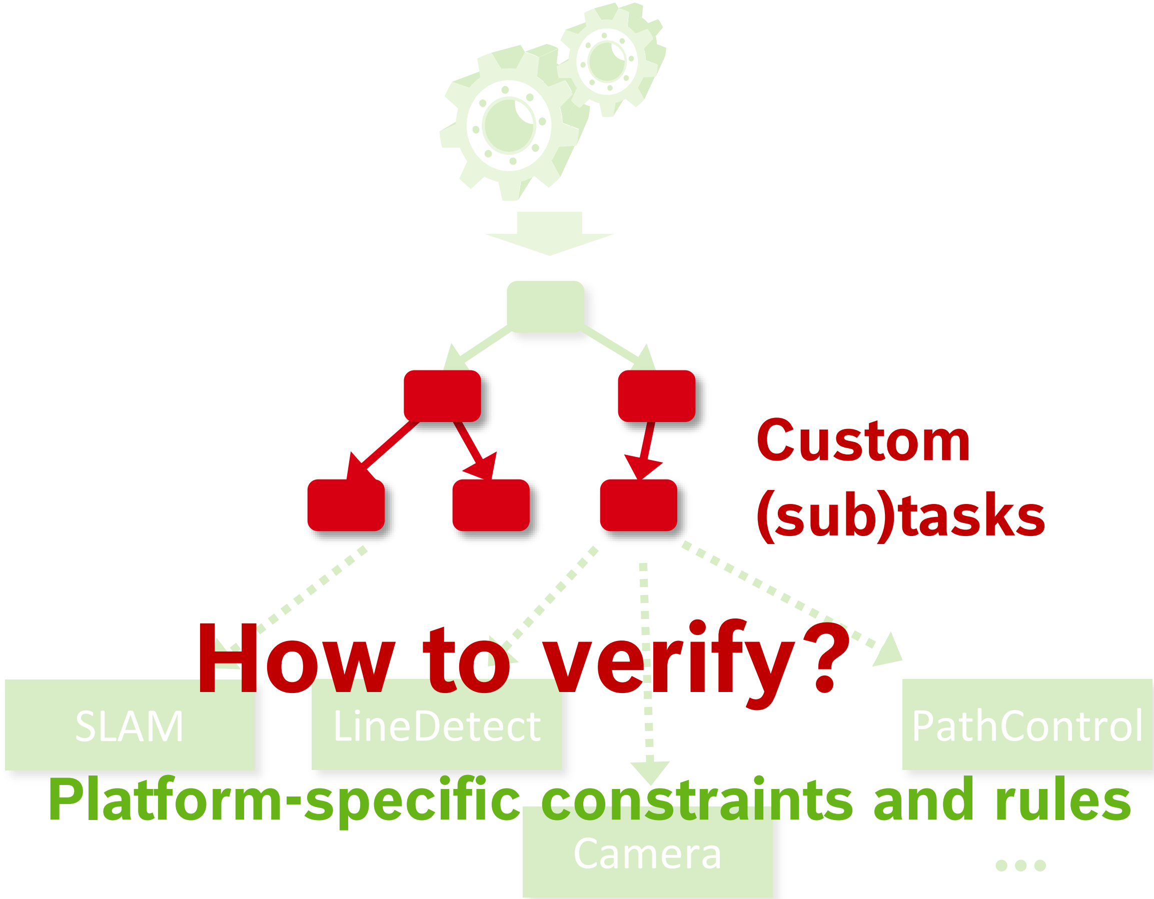
CR/AEE4-Heinzemann, CR/AEE1-Lange | 2018-10-04

# vTSL – A Formally Verifiable DSL for Specifying Robot Tasks
## Motivation

Planning

Executive

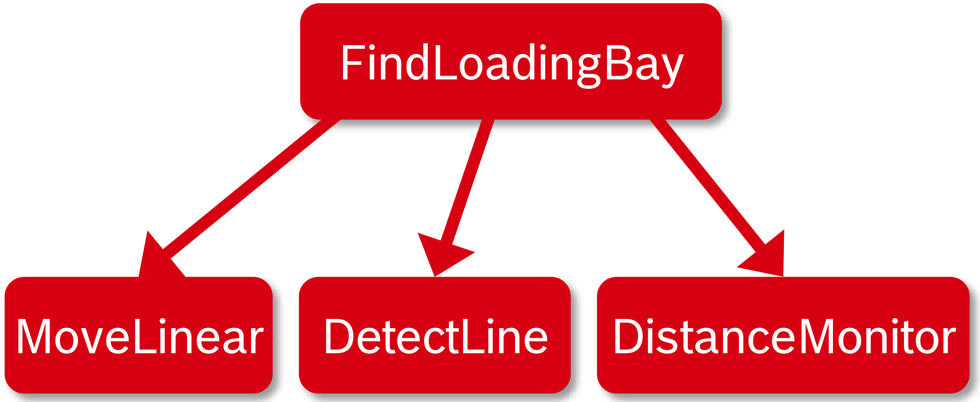
Skills






CR/AEE4-Heinzemann, CR/AEE1-Lange | 2018-10-04

# vTSL – A Formally Verifiable DSL for Specifying Robot Tasks
## Language Concepts and Example

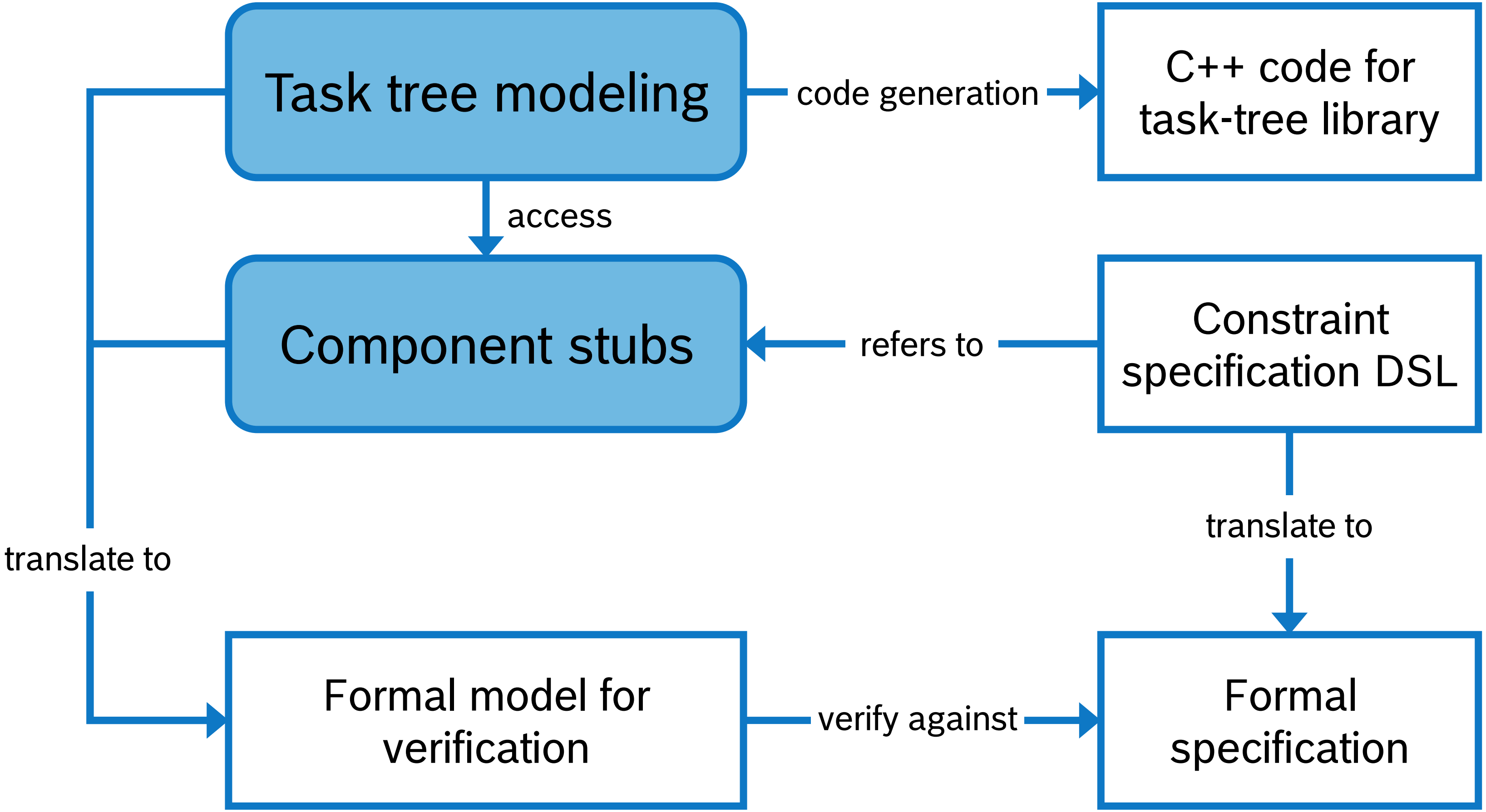



```
while(retries < maxRetries)
  [Status moveStatus, Status lineStatus, Status distStatus] = call/or*
     MoveLinear(speed), DetectLine(<< ... >>), DistanceMonitor(100);
  if ( lineStatus == success ) then
    setGroundCameraState(false);
    return << ... >>;
  else if ( distStatus == success ) then
    // failed to detect line -> drive back and try again
    speed = -speed;
  end
  retries++;
end

setGroundCameraState(false);
```

Task-tree semantics · C-like expressiveness · synchronous programming
interfacing with skill-layer (ROS) · source code generation · verifiability

# vTSL – A Formally Verifiable DSL for Specifying Robot Tasks
## Translation and Model Checking

Implemented using
- JetBrains MPS
- Spin (Promela)





CR/AEE4-Heinzemann, CR/AEE1-Lange | 2018-10-04

# vTSL – A Formally Verifiable DSL for Specifying Robot Tasks

Christian Heinzemann and Ralph Lange

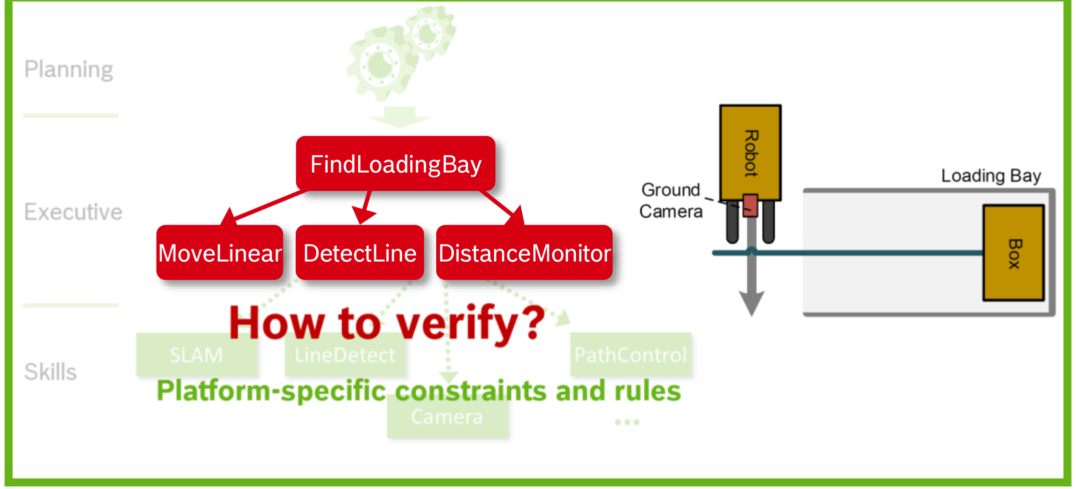
Robert Bosch GmbH
Corporate Sector Research and Advance Engineering
Renningen, Germany
[firstname.lastname]@de.bosch.com

Detailed, poster-like slide set for interactive session

BOSCH

# vTSL – A Formally Verifiable DSL for Specifying Robot Tasks

## Motivation




## Requirements and Key Ideas

**Verifiability**

Fully automated transformation for verification tool – omit feature if not translatable

**Expressiveness**

Modern programming language features desired – resemblance with C/C++

**Interfacing with skill layer**

Interface with ROS and provide abstract replacement model for skill component behaviors

**Task tree semantics**

Principle of decomposition as in Task Description Language or Hierarchical Task Networks

**Synchronous Programming**

Concurrency inspired by Céu, but not the strong determinism of synchronous languages

**Source code generation**

Task trees should be directly generatable into implement-tation for the robot

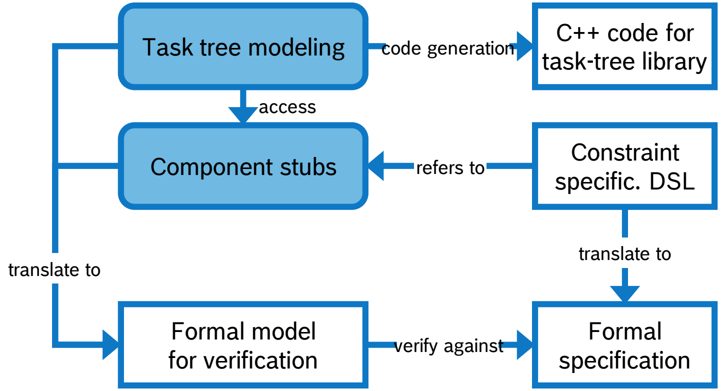
## Task Tree Modeling

```
while(retries < maxRetries)
  [Status moveStatus, Status lineStatus, Status distStatus] = call/or*
     MoveLinear(speed), DetectLine(<< ... >>), DistanceMonitor(100);
  if ( lineStatus == success ) then
    setGroundCameraState(false);
    return << ... >>;
  else if ( distStatus == success ) then
    // failed to detect line -> drive back and try again
    speed = -speed;
  end
  retries++;
end

setGroundCameraState(false);
fail;
```

## Translation and Model Checking

Implemented using

- JetBrains MPS
- Spin (Promela)





CR/AEE4-Heinzemann, CR/AEE1-Lange | 2018-10-04

BOSCH

## Related Work

- A. Nordmann, et al.: "A Survey on Domain-Specific Modeling and Languages in Robotics," JOSER, no. 7, 2016.
  - Most (59 of 137) DSLs have control and event handling on architectural level on focus
  - http://corlab.github.io/dslzoo/
- R. Simmons, et al.: "Towards automatic verification of autonomous systems," IROS 2000.
  - The only verification approach for task trees – but only for activation and synchronization
- A. Cowley and C. J. Taylor, "Towards language-based verification of robot behaviors," IROS 2011.
  - Script language for programming manipulation tasks
  - Supports semiautomatic verification with Coq
- Various verification approaches for (hierarchical) FSMs from CPS community and robotics



BOSCH

# vTSL – A Formally Verifiable DSL for Specifying Robot Tasks

## Verifiability

Fully automated transformation for verification tool – omit feature if not translatable

## Task tree semantics

Principle of decomposition as in Task Description Language or Hierarchical Task Networks

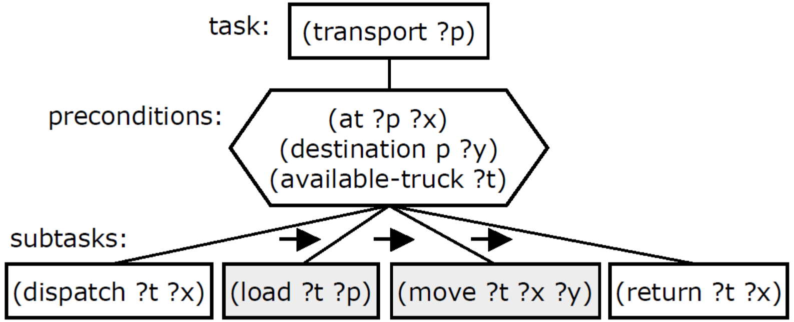



Image taken from Dana Nau et al.: “SHOP2: An HTN Planning System”, *Journal of Artificial Intelligence Research*, Vol. 20, pp. 379-404, 2003.

## Expressiveness

Modern programming language features desired – resemblance with C/C++

## Synchronous Programming

Concurrency inspired by Céu, but not the strong determinism of synchronous languages

```
par/or do
    await RETRANSMIT;
with
    par/and do
        await 1min;
    with
        <send-beacon-packet>
    end
end
```

Snippet from Francisco Sant'Anna: “Structured Synchronous Reactive Programming with Céu”, *Proc. of 14th MODULARITY*, pp. 29-40, 2015.

## Interfacing with skill layer

Interface with ROS and provide abstract replacement model for skill component behaviors

## Source code generation

Task trees should be directly generatable into implement-tation for the robot




BOSCH

# vTSL – A Formally Verifiable DSL for Specifying Robot Tasks

## Action FindLoadingBay

```
action FindLoadingBay(<< ... >>)

  behavior normal
    setGroundCameraState(true);
    uint8 const maxRetries = 3;
    uint8 retries = 0;
    double speed = 0.2;

    while(retries < maxRetries)
      [Status moveStatus, Status lineStatus, Status distStatus] = call/or*
        MoveLinear(speed), DetectLine(<< ... >>), DistanceMonitor(100);
      if ( lineStatus == success ) then
        setGroundCameraState(false);
        return << ... >>;
      else if ( distStatus == success ) then
        // failed to detect line -> drive back and try again
        speed = -speed;
      end
      retries++;
    end

    setGroundCameraState(false);
    fail;
  end

  function setGroundCameraState(boolean state)
    SetBool srvQuery;
    srvQuery.Request.data = state;
    query GroundCamera.activateCamera(srvQuery);
  end

end
```

## Action DetectLine

```
action DetectLine(<< ... >>)

  behavior normal
    connect GroundCamera.lineDetectTopic as cameraImg with queue size 1;
    LineDetectMsg curMsg = read cameraImg;
    while(curMsg.blueComponent < 60)
      curMsg = read cameraImg;
    end on abort
      disconnect cameraImg;
    end
    disconnect cameraImg;
    return << ... >>;
  end

end
```

## Action MoveLinear

```
action MoveLinear(double speed)

  behavior forward execute if speed >= 0
    Float64 speedMsg;
    connect RobotBase.executed as speedMsgExec with queue size 1;
    while(true)
      speedMsg.data = speed;
      write speedMsg to RobotBase.linearSpeed;
      // wait for robot base controller
      read speedMsgExec;
    end
  end on abort
    Float64 speedMsg;
    speedMsg.data = 0.0;
    write speedMsg to RobotBase.linearSpeed;
  end

  behavior backwards execute if speed < 0
    assert(speed > -0.1);
    // ... enable signaling ...
       ... send speed msg to robot base ...
  end

end
```

Formula to be verified by the model checker


CR/AEE4-Heinzemann, CR/AEE1-Lange | 2018-10-04

# vTSL – A Formally Verifiable DSL for Specifying Robot Tasks

## ROS Message

```
@ROS Message, Package Resource Name: iros_robot
import Image

struct LineDetectMsg {
  Image rawImage;
  uint8 blueComponent;
}
```

## ROS Service

```
@ROS Service, Package Resource Name: std_srvs/SetBool

service SetBool {
  struct Request {
    boolean data; // e.g. for hardware enabling / disabling
  }

  struct Response {
    boolean success; // indicate successful run of triggered service
    string message; // informational, e.g. for error messages
  }
}
```

## Component Stub

```
skill component GroundCamera {

  Advertised Topics:
    topic LineDetectMsg lineDetectTopic;

  Offered Services:
    service SetBool activateCamera;

  initialization
    boolean cameraState = false;
    SetBool.Request activateCamQuery;
    SetBool.Response activateCamResp;
  end

  execute behavior
    choose
      [activateCamQuery, activateCamResp] = receive GroundCamera.activateCamera();
      cameraState = activateCamQuery.data;
      activateCamResp.success = true;
      reply GroundCamera.activateCamera(activateCamResp, true);
    or
      (cameraState == true);
      LineDetectMsg lineMsg;
      choose
        lineMsg.blueComponent = 59;
      or
        lineMsg.blueComponent = 61;
      end
      write lineMsg to GroundCamera.lineDetectTopic;
    end
  end
}
```


CR/AEE4-Heinzemann, CR/AEE1-Lange | 2018-10-04

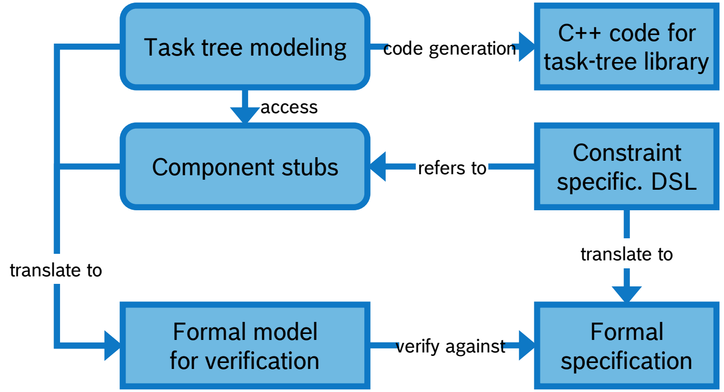

BOSCH

# vTSL – A Formally Verifiable DSL for Specifying Robot Tasks

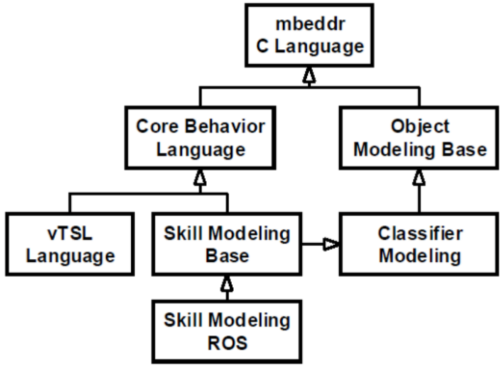
## Translation and Model Checking

Implemented using

- JetBrains MPS
- Spin (Promela)




## Language Dependencies




## Promela Model

```
byte runningAction;
byte readyQueue[DISPATCHER_QUEUE_SIZE];
byte blockedQueue[DISPATCHER_QUEUE_SIZE];
byte abortSubaction[2];

proctype FindLoadingBay(){
  byte childPid, childPid2;
  { //wait for run signal
    (runningAction == _pid);

    //user defined action code starts here!
    _FindLoadingBay_setGroundCameraState(true);
    #define maxRetries 3
    byte retries = 0;

    do
    :: retries < maxRetries -> {
       mtype distStatus;
       childPid = run gen__lambda__IROS_Gen_FindLoadingBay_1();
       childPid2 = run IROS_Gen_DistanceMonitor(100);
       callParallelSubactions(childPid, childPid2);
       awaitParallelORSubactionReturn(childPid, childPid2, _, distStatus);
       (runningAction == _pid);
       if
       :: IROS_Gen_FindLoadingBay_normal_lineStatus == SUCCESS -> {
          _IROS_Gen_FindLoadingBay_setGroundCameraState(false);
          ActionReturn(SUCCESS); }
       :: else -> { ... }
       fi;
       retries++; }
    :: else -> { break; }
    od;

    _FindLoadingBay_setGroundCameraState(false);
    ActionReturn(FAILED);
    //user defined action code ends here!
  } unless {
    {(abortSubaction[0] == _pid || abortSubaction[1] == _pid);
    (runningAction == _pid);
      ActionReturn(ABORTED);
    };
  };
  ActionReturn(SUCCESS);
}
```


CR/AEE4-Heinzemann, CR/AEE1-Lange | 2018-10-04

# vTSL – A Formally Verifiable DSL for Specifying Robot Tasks

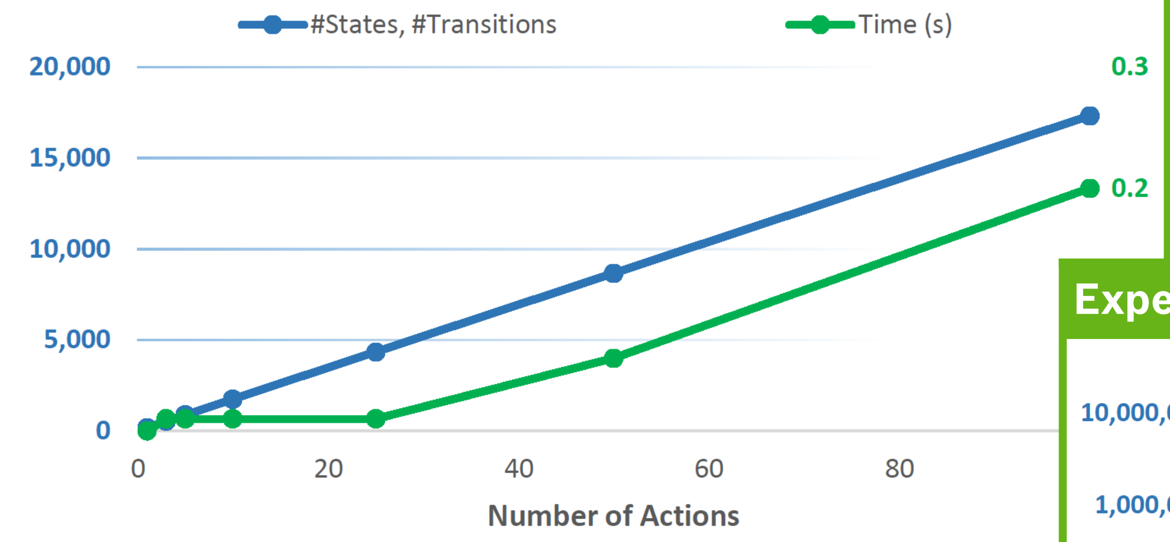
PML file

The Spin Model Checker (spinroot.com)

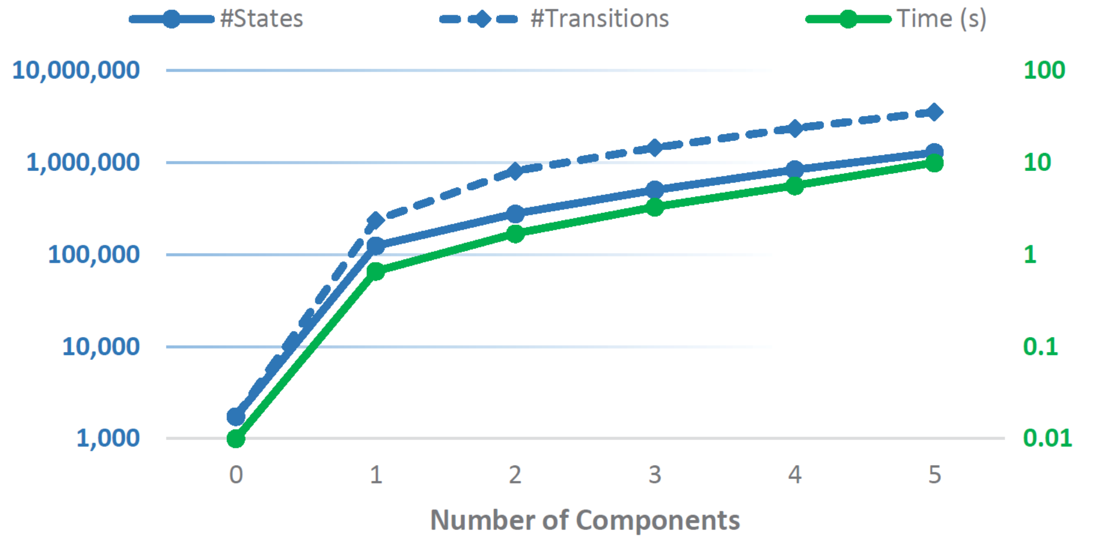
```
$ spin -search -i MyTaskTree.pml
pan:1: assertion violated (BenchmarkComponent_simpleTopic.registeredClients<1) (at depth 286)
pan: wrote MyTaskTree.pml.trail
pan: reducing search depth to 287

(Spin Version 6.4.8 -- 2 March 2018)
  + Partial Order Reduction

Full statespace search for:
  never claim           - (none specified)
  assertion violations  +
  cycle checks          - (disabled by -DSAFETY)
  invalid end states  +

State-vector 668 byte, depth reached 286, errors: 1
       97 states, stored
        7 states, matched
      104 transitions (= stored+matched)
      212 atomic steps
hash conflicts:         0 (resolved)

unreached in proctype vtsl__dispatcher
  MyTaskTree.pml:618, state 8, "abortRootAction = 0"
  MyTaskTree.pml:617, state 9, "abortSubaction[0] = rootActionPid"
  MyTaskTree.pml:624, state 16, "-end-"
  (3 of 16 states)
unreached in proctype vtsl__externalEventHandler
  MyTaskTree.pml:240, state 8, "vtsl__i = (vtsl__i+1)"
  MyTaskTree.pml:247, state 16, "vtsl__i = 0"

                 ...

  MyTaskTree.pml:1112, state 106
  MyTaskTree.pml:1120, state 110, "-end-"
  (24 of 110 states)

pan: elapsed time 0.02 seconds
pan: rate      4850 states/second
```

# vTSL – A Formally Verifiable DSL for Specifying Robot Tasks

## Experiment 1: Scalability w.r.t. Tree Size



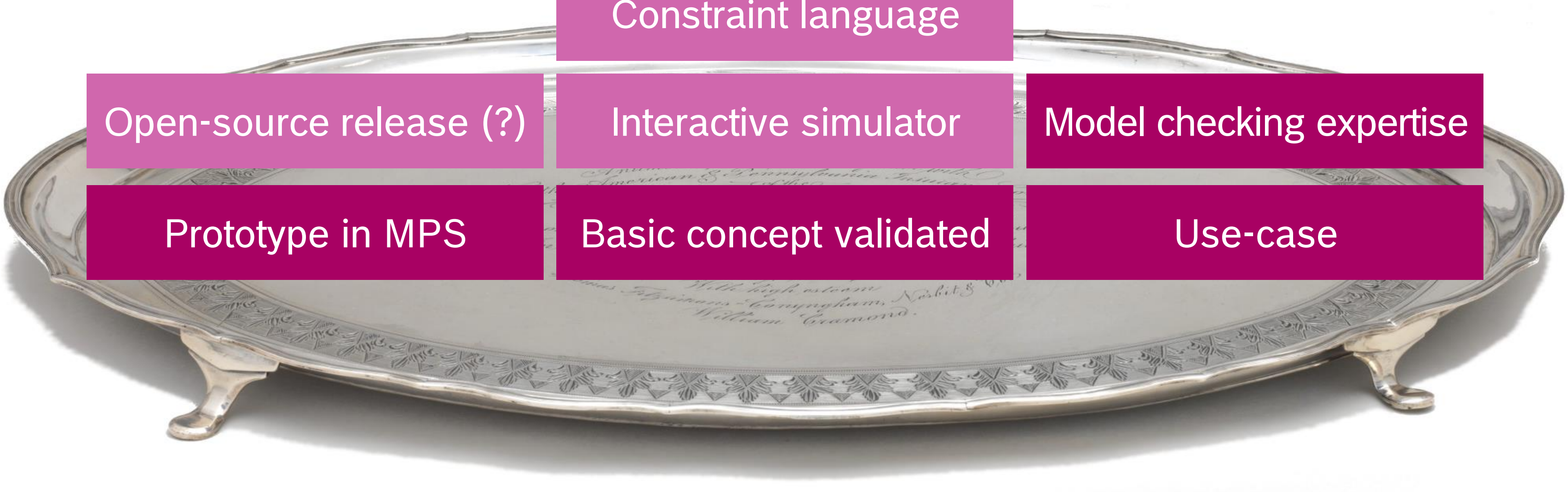

## Experiment 2: Scalability w.r.t. Component Count





CR/AEE4-Heinzemann, CR/AEE1-Lange | 2018-10-04

# vTSL – A Formally Verifiable DSL for Specifying Robot Tasks



Environment model

Constraint language

Open-source release (?)

Interactive simulator

Model checking expertise

Prototype in MPS

Basic concept validated

Use-case

**We are seeking out for an academic partner to join forces!**



BOSCH