Robots everywhere:

the next step after PCs ?
with MSRS ?

Bernhard Merkle
Central Unit Research & Development
Software-Engineering
SICK-AG Waldkirch

mailto: Bernhard.Merkle@sick.de

mailto: Bernhard.Merkle@gmail.com
Contact on linkedin.com or xing.com

Overview:

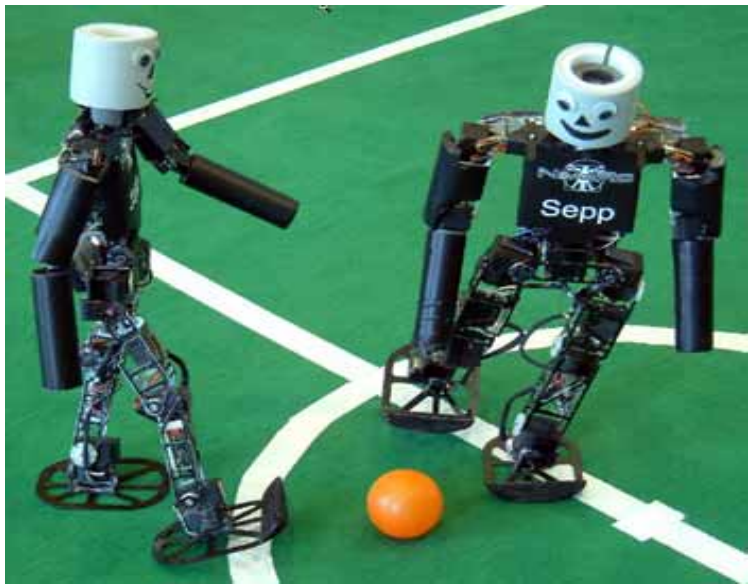
- Robotics
  - Definitions, Potentials, Challenges
- Microsoft Robotics Studio (MSRS)
  - MSRS Architecture
  - Runtime
    - Distributed System Services, DSS
    - Concurrency and Coordination Runtime, CCR
  - Authoring Tools
    - Development Tools,
    - Visual Programming Language VPL
  - Services and Samples
    - Robot Services and Models
    - Documentation, Community
- Other Robotics Toolkits
  - Open Source, Lego (RIS/NXT), FischerTechnik, etc
- Demos

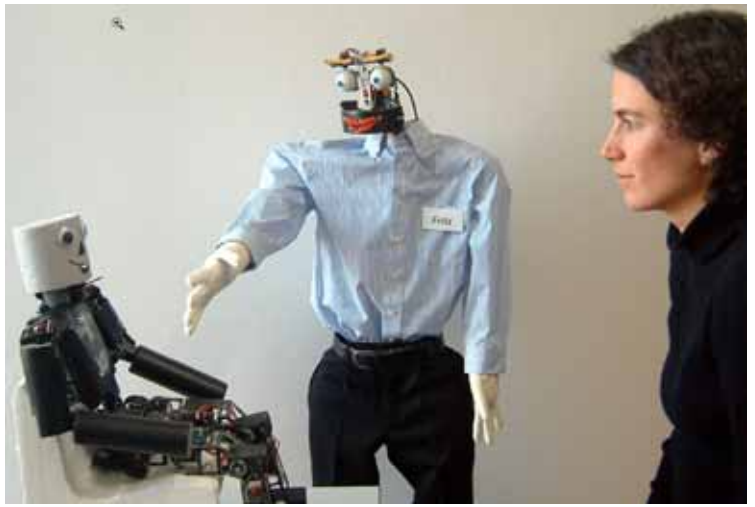
Trial of Definition: A Robot is

SICK

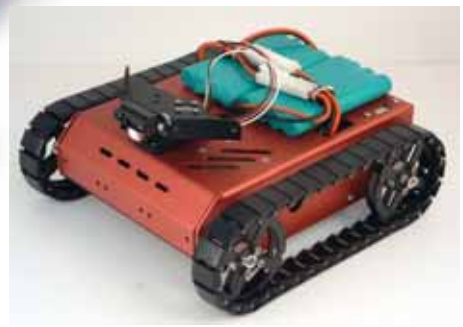
- device,
- hard- and/or software
- with the capability of sensing
- and (re-)acting

- Has to react to the world →
  - Concurrent
  - Parallel
  - Have/Adopt some intelligence





Highly Diverse Market…

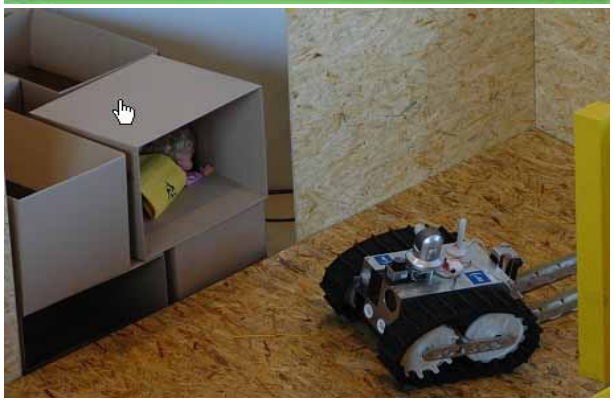


RoboCup (by 2050 we will lose against robots ☺)

SICK

RoboCup Soccer

- Simulation
- Small size
- Middle size
- Humanoid

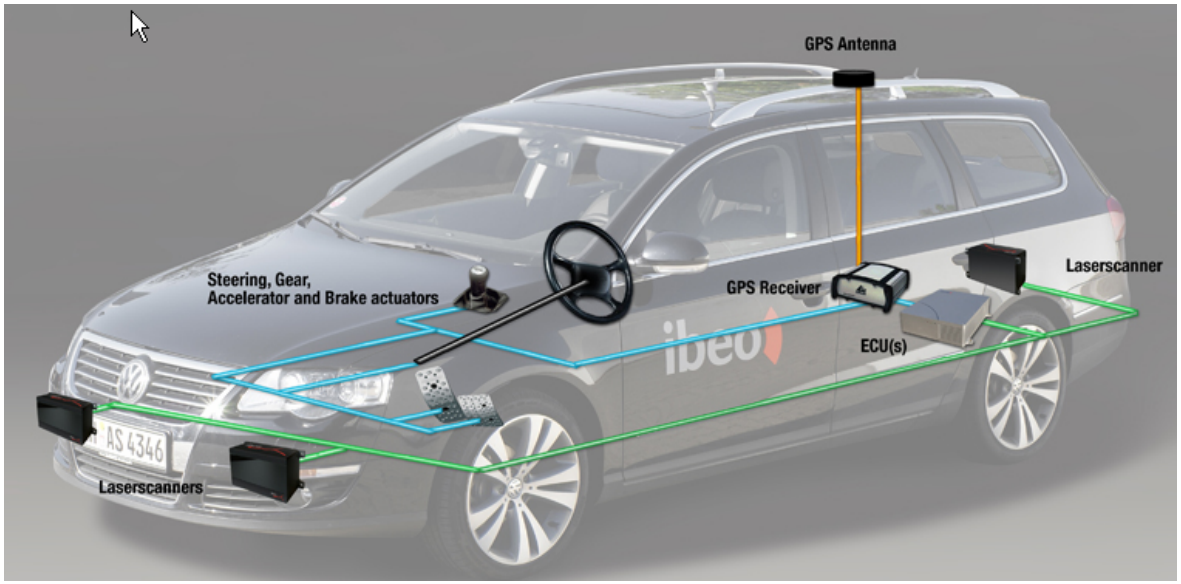


RoboCup Rescue

- Simulation
- Rescue Robots

DARPA (05/07): autonomous vehicles…

SICK



http://pave.princeton.edu/main/

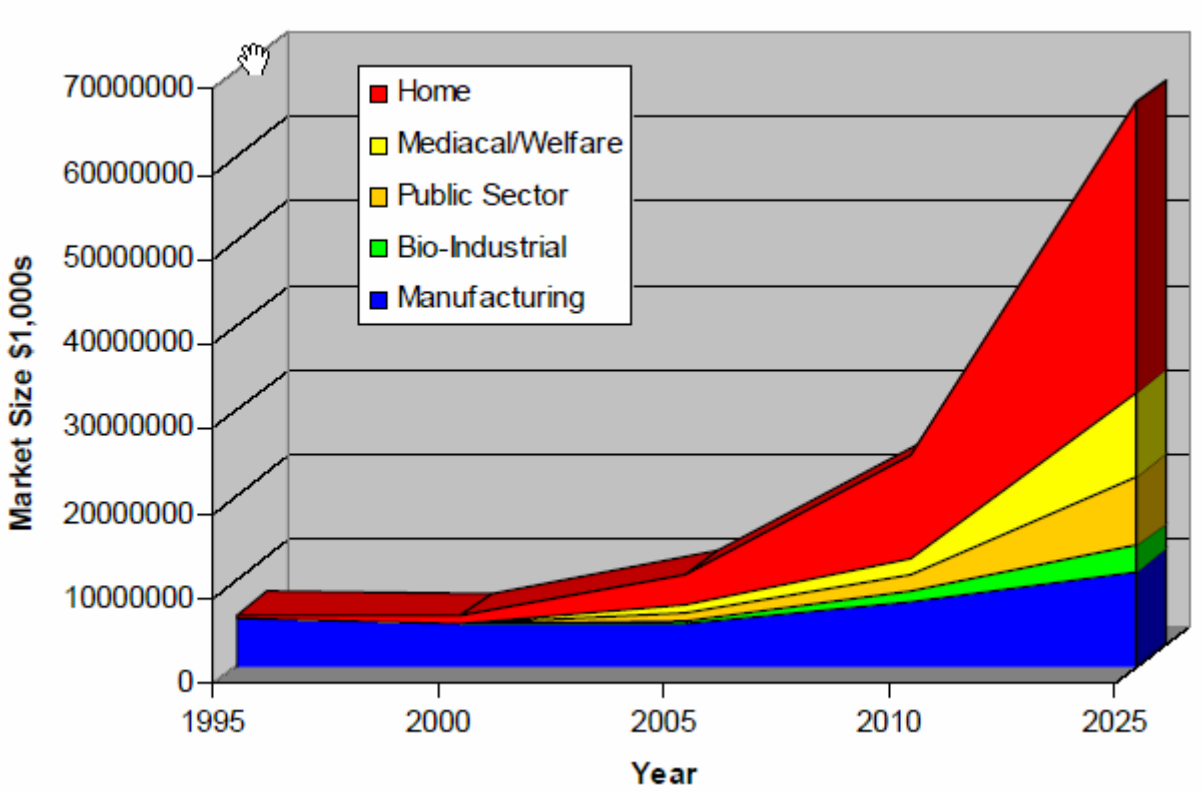
Robotics Market Potential / Challenge

Emerging service and consumer market

- Remote assistance/presence
- Facilities maintenance
- Security
- Education
- Entertainment

Emerging challenges

- Lack of reusability
- Reinventing
- Concurrency
- Complexity



*Source: Japan Robotics Association

M$ aims with MSRS (M$ Robotics Studio)

SICK

Usage

- Primary market:
  - Endconsumer / Personal-Robotics (2nd wave of PC)
- Secondary market:
  - concurrent programming (next “wave” of programming)
  - CCR (will be) distributed separately
  - Coordination with MS “Parallel Computing” Initiative (Parallel Extensions to .NET Framework CTP)

Team

- Tandy Trower
- George Chrysanthakopoulos
- Henrik F Nielsen
- Small team (approx. 15) of high educated staff
- Kind of Startup in Microsoft

Robotics: Shared Challenges…

SICK

Input from industry, hobbyists, academia, research,…

- Configuring sensors and actuators in running system
- Coordinating sensors and actuators asynchronously
- Starting and stopping components dynamically
- Monitoring/Interacting/Debugging running system
- Development when access to robot is limited
- Span multiple compute units
- Re-use of components across hardware platforms and devices

What Microsoft

learned…

- Too much complexity
- Too many resources required
- Lack of reusability, choice
- Limited tools and technologies
- Difficulties in sharing
- Transference of skills/experience

intends… ;-)

- Next Wave after PCs will be robots !?
- “A Robot in Every Home”, Bill Gates in AS
- Be(come) a market leader for Robotics Solutions
- Establish .NET Technology also in Industry ?

SICK

Addressing the Challenges

Runtime environment

- Execute, monitor, and interact with robotics applications

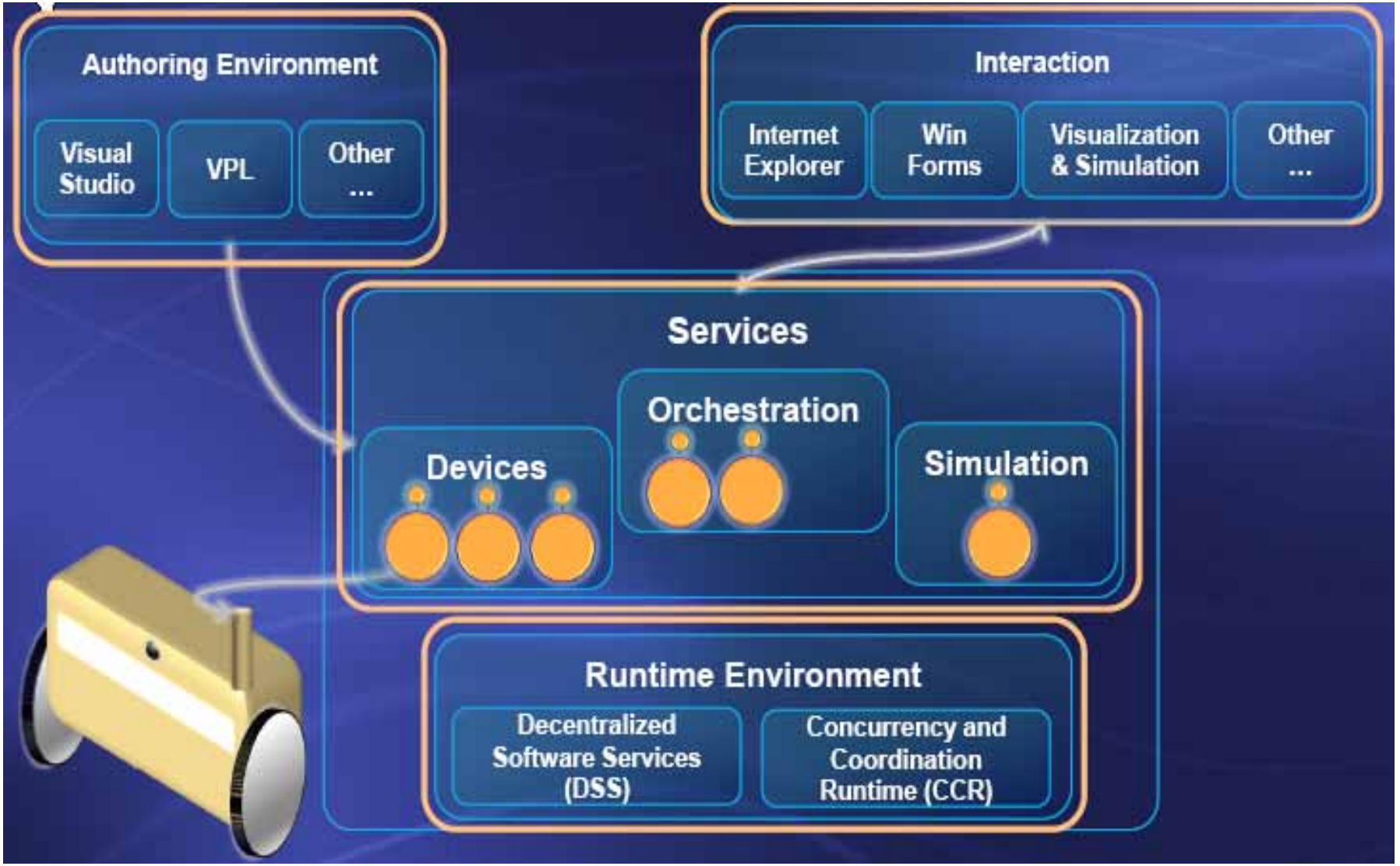
Authoring environment

- Write, orchestrate, and deploy robotics applications

Simulation environment

- Execute robotics applications using simulated hardware, physical entities, and terrain

Microsoft Robotics Studio: High-level Architecture



Microsoft Robotics Studio

SICK

A **lightweigth concurrency** and **service oriented** runtime

- Handling sensor input / controlling actuators
- Based on message passing
- DSS facilitating tasks and basic services

Authoring/development tools

- Visual programming editor (VPL)
- Visual Studio IDE (.NET languages)
- Message debugging
- Simulation

Libraries and basic algorithms

- Code samples
- Documentation

Microsoft Robotics Studio

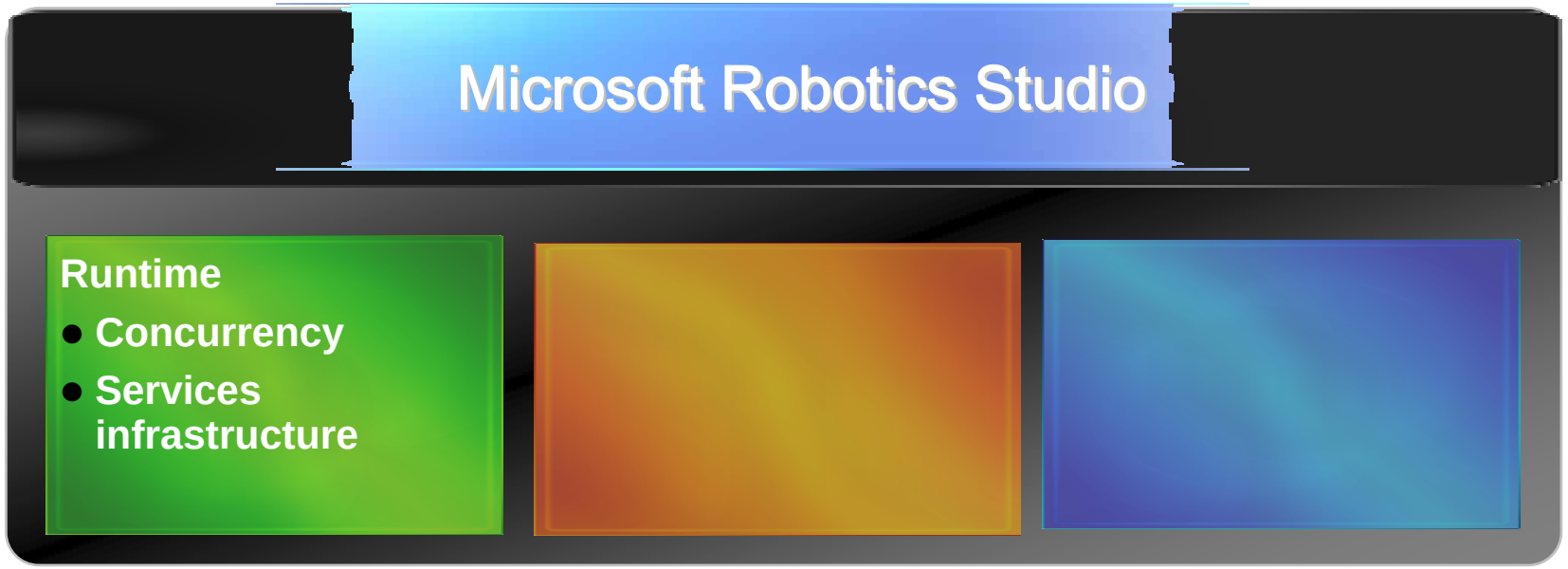
SICK

A development platform for the robotics community, supporting a wide variety of users, hardware, and application scenarios

Microsoft Robotics Studio

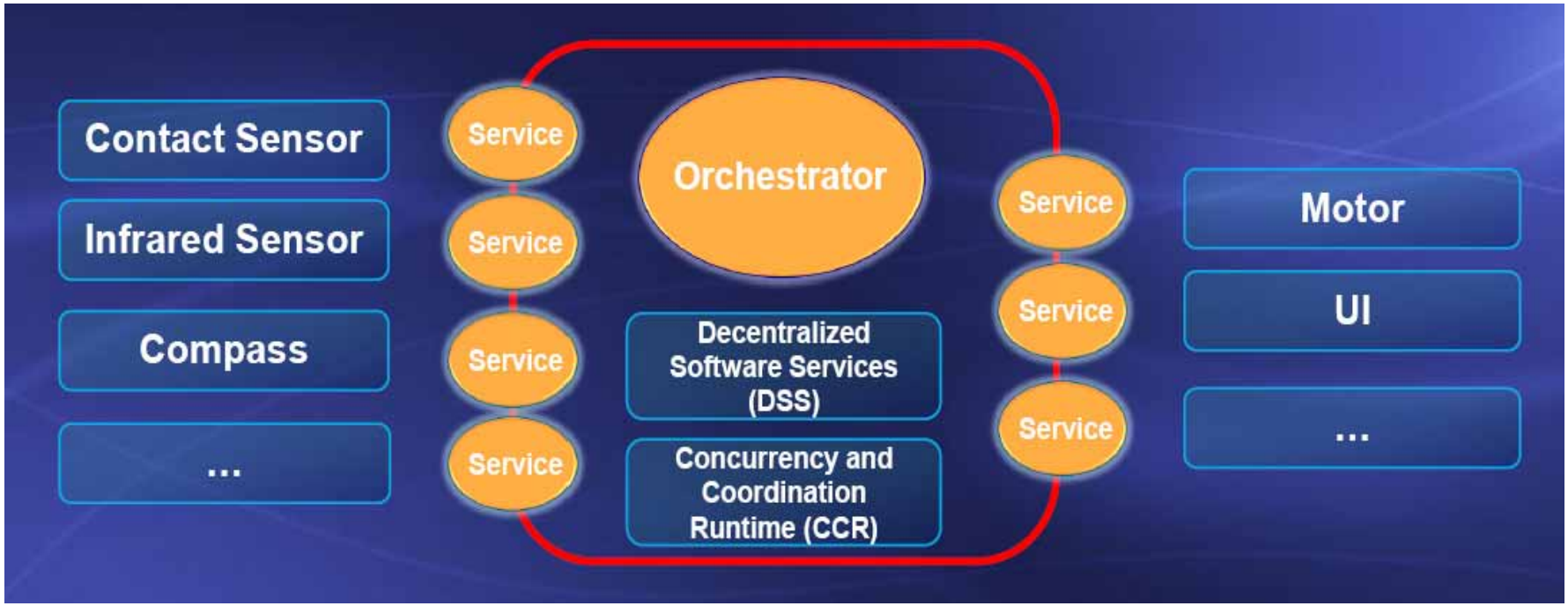
SICK

A development platform for the robotics community, supporting a wide variety of users, hardware, and application scenarios



Application Model

SICK



An application is a **composition** of loosely-coupled components **concurrently** executing

- Orchestration is a service
- Partnering and subscription (static, dynamic)
- Services are units of orchestrations (nested)

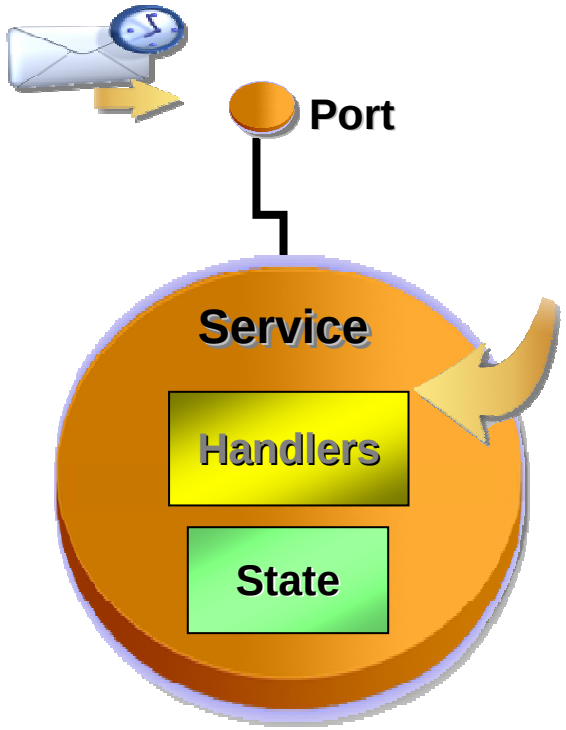
Service: simple abstraction for hard-/software

SICK

Service properties:

- Identity
- Sturctured State
- Operations and Handlers
- Support dynamic discovery
- Distributed and asynchronous

→ Service Contract



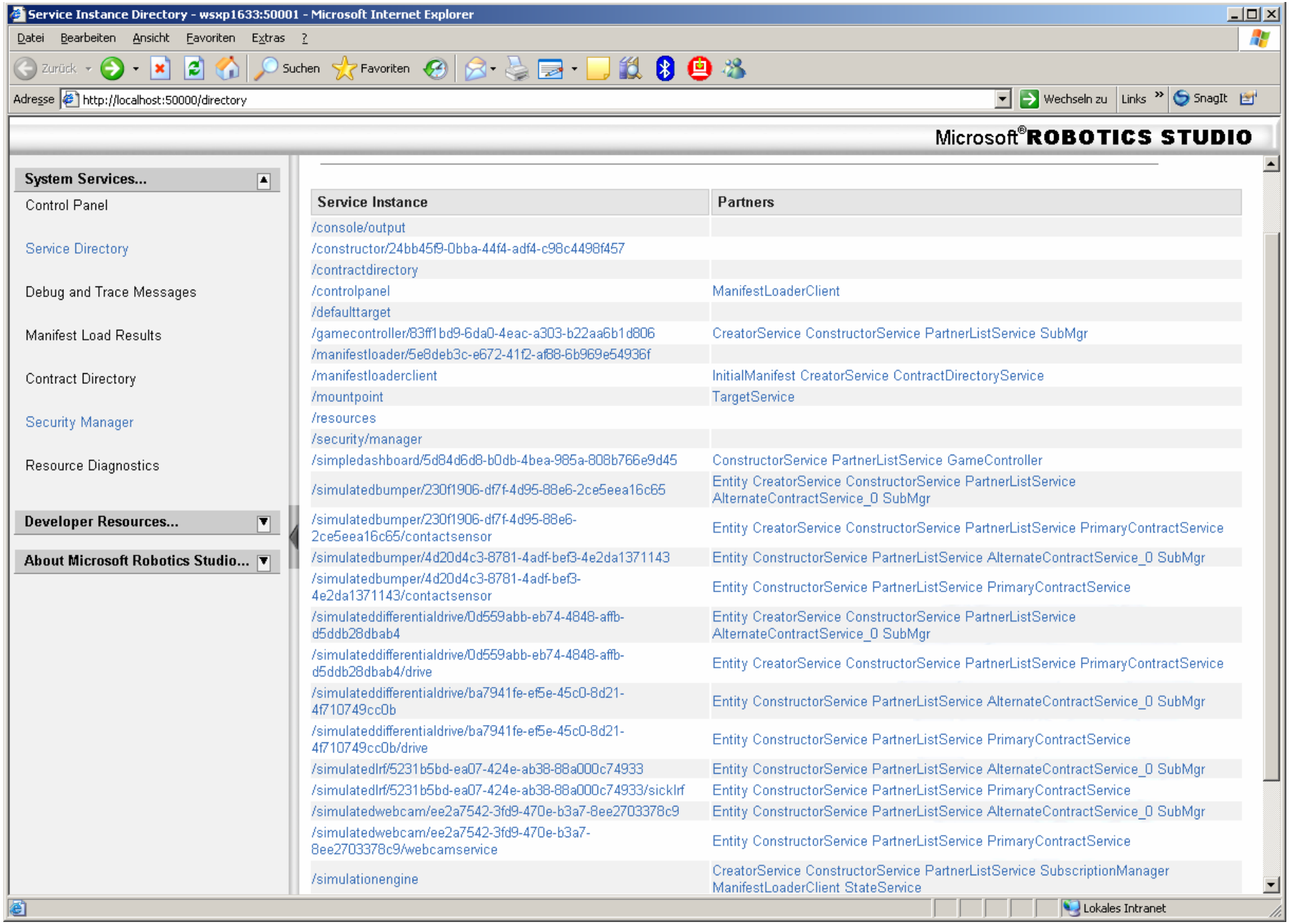
Uniform behaviour

- State retrieval / manipulation
- Service creation / termination
- Event notification
- Re-use (composition, aggregation)

SICK

Protocol based service interaction

- DSSP is a SOAP-base protocol for inter-service communication
  - Based on HTTP (existing web infrastructure)
  - Structured data manipulation,
  - Event notification
  - TCP or HTTP for Intra-node
  - open / performs


Service Instance Directory - wsxp1633:50001 - Microsoft Internet Explorer
Datei Bearbeiten Ansicht Favoriten Extras ?
Zurück Suchen Favoriten
Adresse http://localhost:50000/directory
Wechseln zu Links SnagIt
Microsoft® ROBOTICS STUDIO
System Services...
Control Panel
Service Directory
Debug and Trace Messages
Manifest Load Results
Contract Directory
Security Manager
Resource Diagnostics
Developer Resources...
About Microsoft Robotics Studio...
Service Instance
Partners
/console/output
/constructor/24bb45f9-0bba-44f4-adf4-c98c4498f457
/contractdirectory
/controlpanel ManifestLoaderClient
/defaulttarget
/gamecontroller/83ff1bd9-6da0-4eac-a303-b22aa6b1d806 CreatorService ConstructorService PartnerListService SubMgr
/manifestloader/5e8deb3c-e672-41f2-af88-6b969e54936f
/manifestloaderclient InitialManifest CreatorService ContractDirectoryService
/mountpoint TargetService
/resources
/security/manager
/simpledashboard/5d84d6d8-b0db-4bea-985a-808b766e9d45 ConstructorService PartnerListService GameController
/simulatedbumper/230f1906-df7f-4d95-88e6-2ce5eea16c65 Entity CreatorService ConstructorService PartnerListService AlternateContractService_0 SubMgr
/simulatedbumper/230f1906-df7f-4d95-88e6-2ce5eea16c65/contactsensor Entity CreatorService ConstructorService PartnerListService PrimaryContractService
/simulatedbumper/4d20d4c3-8781-4adf-bef3-4e2da1371143 Entity ConstructorService PartnerListService AlternateContractService_0 SubMgr
/simulatedbumper/4d20d4c3-8781-4adf-bef3-4e2da1371143/contactsensor Entity ConstructorService PartnerListService PrimaryContractService
/simulateddifferentialdrive/0d559abb-eb74-4848-affb-d5ddb28dbab4 Entity CreatorService ConstructorService PartnerListService AlternateContractService_0 SubMgr
/simulateddifferentialdrive/0d559abb-eb74-4848-affb-d5ddb28dbab4/drive Entity CreatorService ConstructorService PartnerListService PrimaryContractService
/simulateddifferentialdrive/ba7941fe-ef5e-45c0-8d21-4f710749cc0b Entity ConstructorService PartnerListService AlternateContractService_0 SubMgr
/simulateddifferentialdrive/ba7941fe-ef5e-45c0-8d21-4f710749cc0b/drive Entity ConstructorService PartnerListService PrimaryContractService
/simulatedirf/5231b5bd-ea07-424e-ab38-88a000c74933 Entity ConstructorService PartnerListService AlternateContractService_0 SubMgr
/simulatedirf/5231b5bd-ea07-424e-ab38-88a000c74933/sicklrf Entity ConstructorService PartnerListService PrimaryContractService
/simulatedwebcam/ee2a7542-3fd9-470e-b3a7-8ee2703378c9 Entity ConstructorService PartnerListService AlternateContractService_0 SubMgr
/simulatedwebcam/ee2a7542-3fd9-470e-b3a7-8ee2703378c9/webcamservice Entity ConstructorService PartnerListService PrimaryContractService
/simulationengine CreatorService ConstructorService PartnerListService SubscriptionManager ManifestLoaderClient StateService
Lokales Intranet

SICK

Asynchronous Programming model

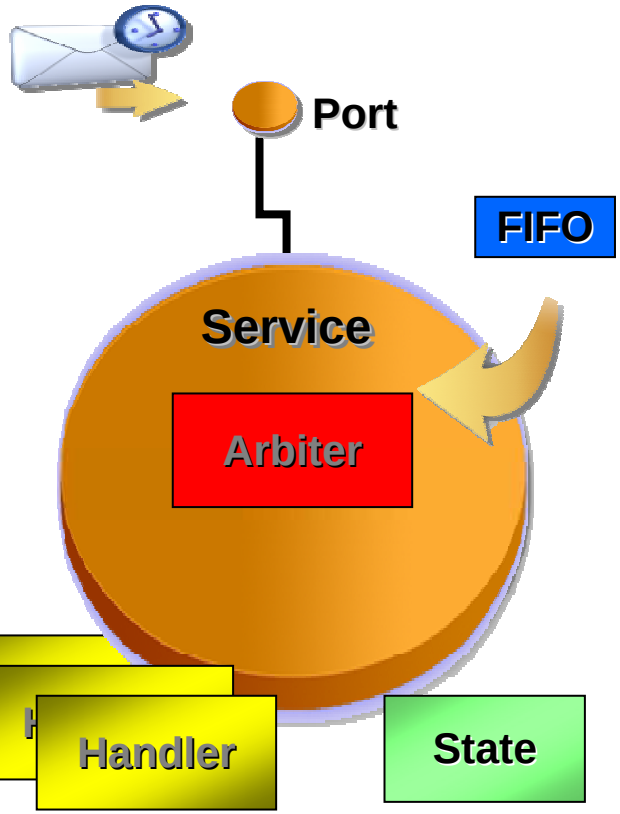
- A concurrency model without manual threading, locks, semaphores, etc.
  - Based on message passing
  - Focus on coordination primitives
  - Sequential execution but
    with no thread blocking, with no need for callback
- Execution context for services
  - Isolation from infrastructure
  - Isolation from other services

Messages, Ports, and Arbiters

SICK

Messages are sent to Ports

```
Port<int> myPort = new Port<int>() ;
myPort.Post (42) ;
```



Ports contain

- A FIFO data-structure holding values in a port
- A list of continuations that can be executed pending message arrival and arbitration
  - A continuation is represented by a C# delegate
  - Can either be named or anonymous

Arbiters

- Implement common concurrency constructs and patterns like
  - choice
  - join
  - interleaved calculations
  - (batch, persistent,…)

CCR Primitives

SICK

Single item receiver

- Executes code when a message arrives

Choice arbiter

- Chooses one receiver (join or single item) from many, across different ports, executes only first one with conditions met, discarding others

Join expressions

- Static join expressions
- Dynamic over a runtime specified number of ports and messages

Interleave arbiter

- Teardown group, Concurrent Group, Exclusive Group

Example: Receive

```
Port<string> port = new Port<string>();
stringPort.Post("StringA");

Arbiter.Activate(
   Arbiter.Receive(stringPort, delegate(string s)
        {Console.WriteLine("Received: " + s);}
));



//Multiple Items
Port<String> stringPort = new Port<String>();
for (int i = 0; i < 50; i++) stringPort.Post(i.ToString());

Arbiter.Activate(
    Arbiter.MultipleItemReceive(stringPort, 10,
                                delegate(String[] strings)
         { Msg("Ten strings={0}", String.Join(", ", strings)); }
));
```

Example: Choice

```
PortSet<int, string> port = new PortSet<int, string>();

Arbiter.Activate(
    Arbiter.Choice(port, MyIntHandler, MyStringHandler)
);

void MyIntHandler(int i)
{
    Console.WriteLine("Received: " + i);
}

void MyStringHandler(string s)
{
    Console.WriteLine("Received: " + s);
}
```

Example: Join

```
Port<double> balancePort = new Port<double>();
Port<int> depositPort = new Port<int>();

Arbiter.Activate(
    Arbiter.JoinedReceive<int,double>(true,
   depositPort, balancePort,
   delegate(int b, double d)
    {
        balance.post(b + d);
    })
);
```

Example: Dynamic Join

```
PortSet<Result,Exception> resultsPort =
   new PortSet<int>();

// parallel computation by posting requests
For (int i=0;i<N;i++)
{
    computePort.Post(new DoWork(someData,resultsPort));
}

// requests complete asynchronously with unknown number
// of failures vs. successes
Arbiter.Activate(
    Arbiter.MultipleItemReceive(resultsPort,
       delegate (ICollection<Result> successes,
                 ICollection<Exception> failures)
    {
        foreach(Result r in results)
        {
            ……
        }
    });
```

Declarative Coordination for Services

SICK

```
[ServiceHandler(ServiceHandlerBehavior.Concurrent)]
public IEnumerator<ITask> GetHandler(Get get)
{
    get.ResponsePort.Post(_state);
    yield break;
}

[ServiceHandler(ServiceHandlerBehavior.Exclusive)]
public IEnumerator<ITask> UpdateHandler(Update update)
{
    _state.CurrentResult += update.Body.Value;
    update.ResponsePort.Post(new UpdateResponse());
    yield break;
}
```

Declarative DataContract for Services

SICK

```
[DataContract]
public class ServiceState
{
        private string _member = "This is my State!";

        [DataMember]
        public string Member
        {
            get { return _member; }
            set { _member = value; }
        }

        private int _ticks;

        [DataMember]
        public int Ticks
        {
            get { return _ticks; }
            set { _ticks = value; }
        }
}
```

Summary: Runtime Environment

SICK

Decentralized Software Services (DSS)

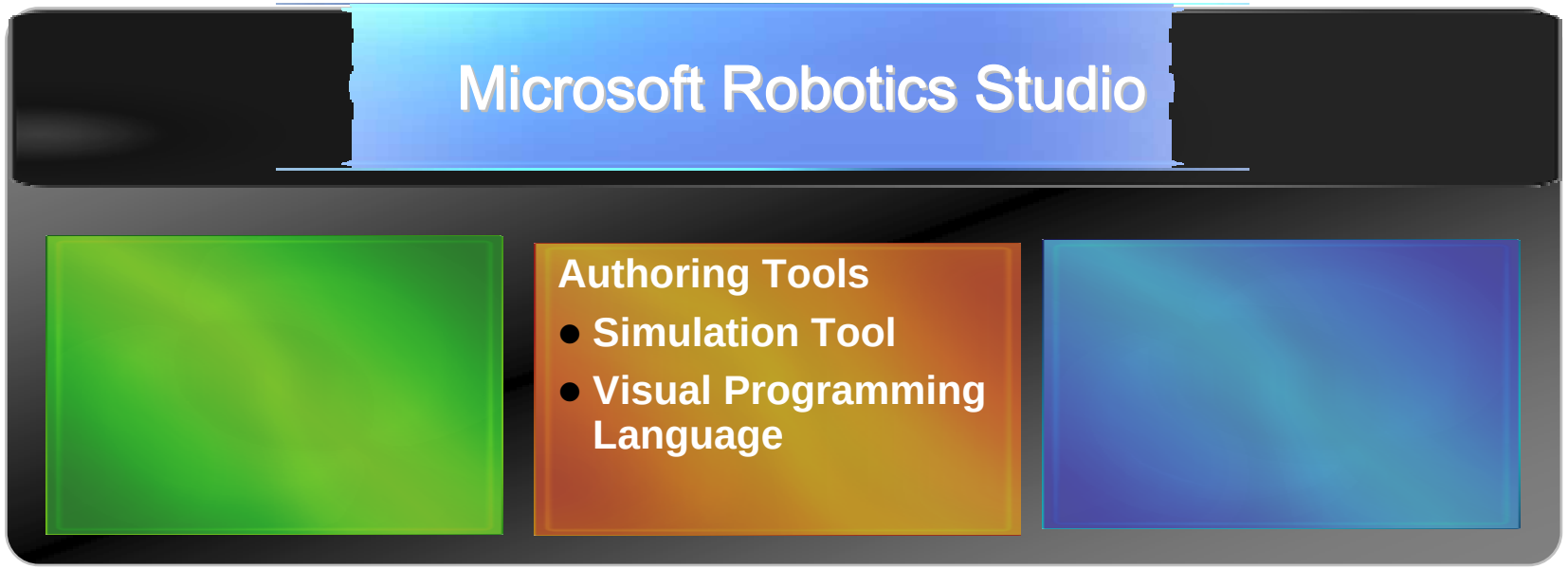
- Easy access: View, access component state
- Flexibility: discover, start, stop services
- Distribuion: transperancy of service location
- Reusable, composable, scalable

Concurrency and Coordination Runtime (CCR)

- Ease of use: small, simple library, that avoids the complexity of manual threading, locks, semaphores, etc
- Concurrency: coordination primitives
  Based on asynchronouse message passing
- Execution context: isolated from infrastructure, other services

Microsoft Robotics Studio

A development platform for the robotics community, supporting a wide variety of users, hardware, and application scenarios



Authoring tools: develop/run a robot application

SICK

Web browser

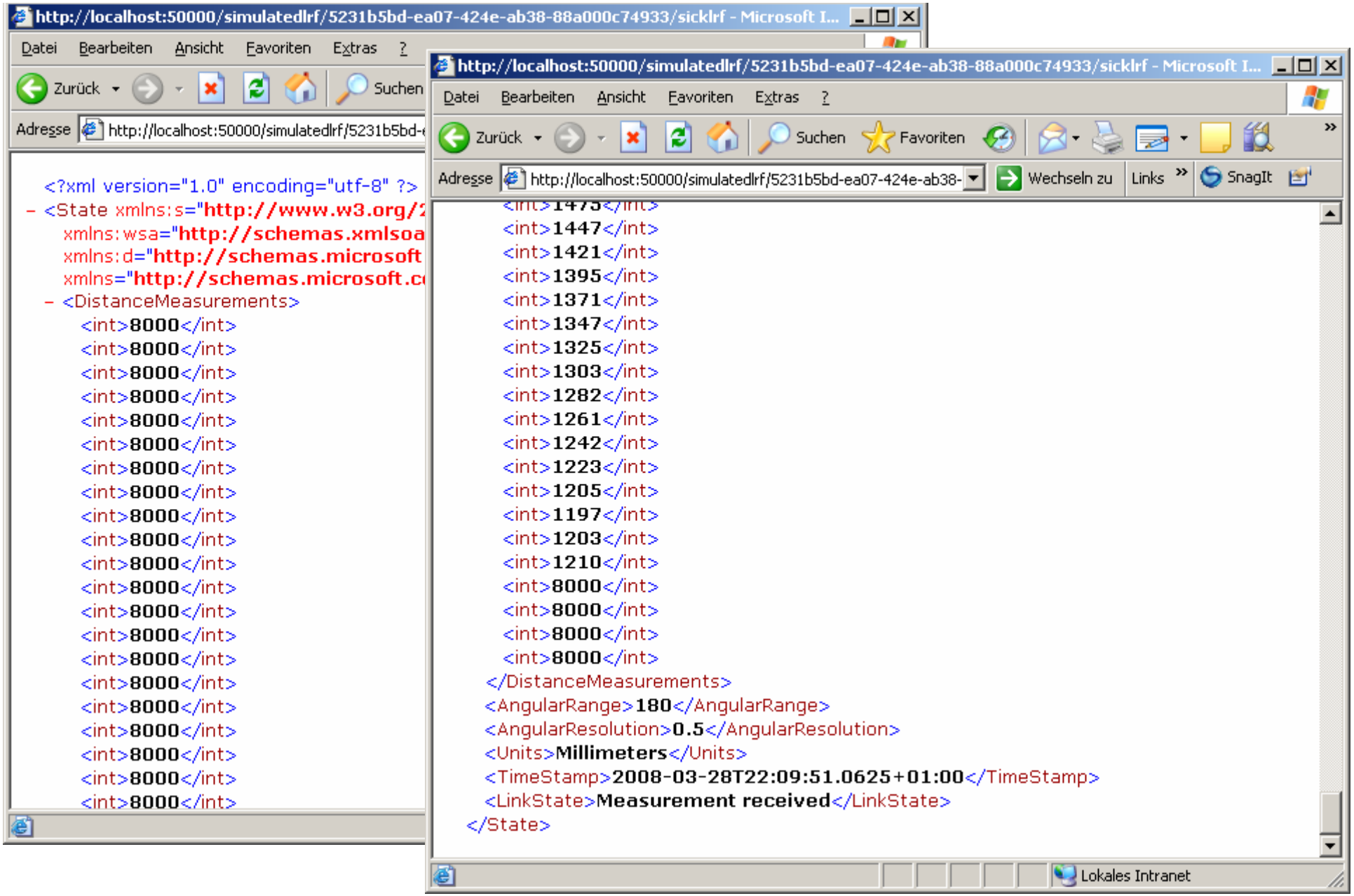
- Inspect/change service state
- Run applications (e.g. JScript as language)

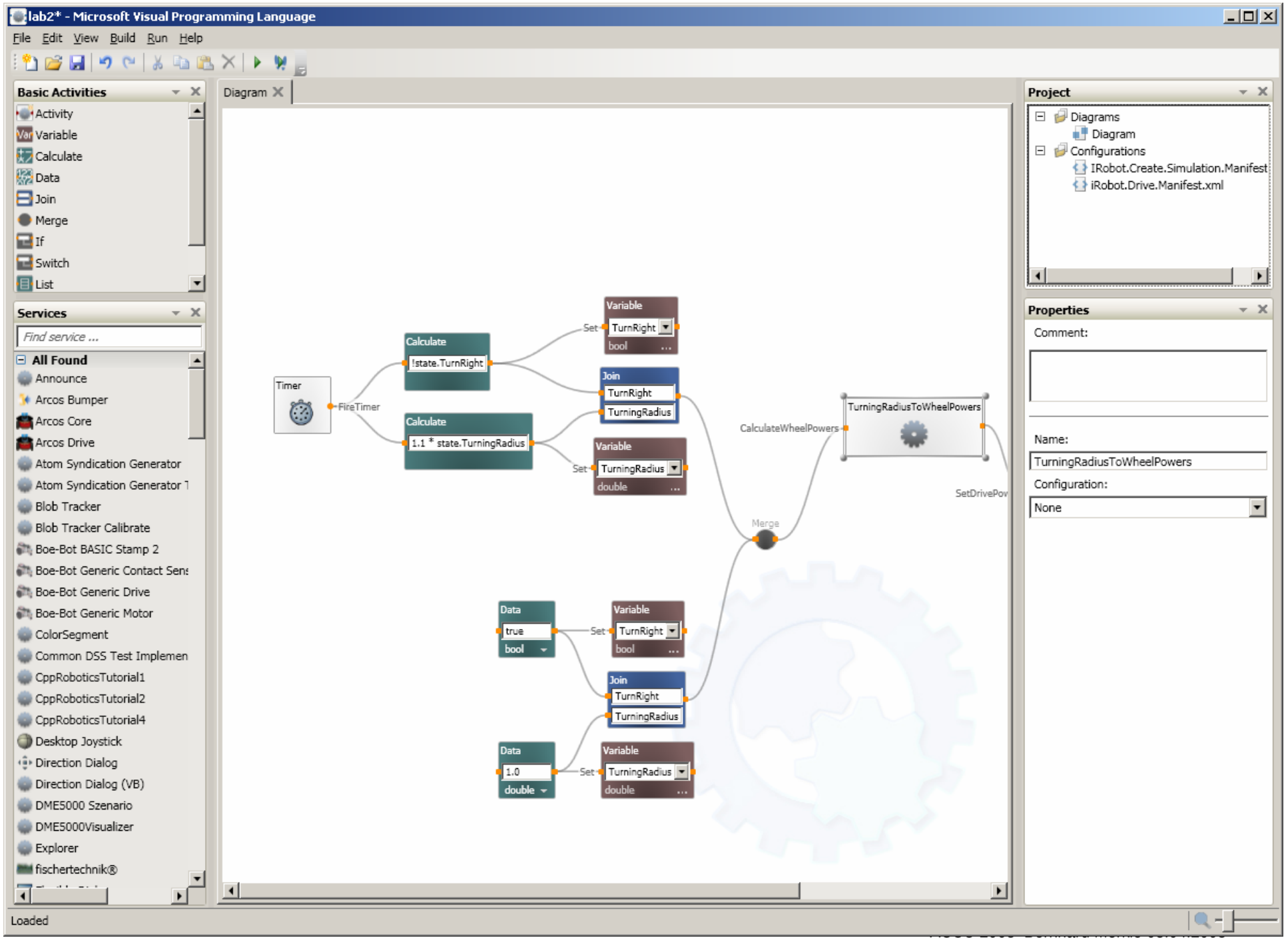
Visual Programming Language (VPL)

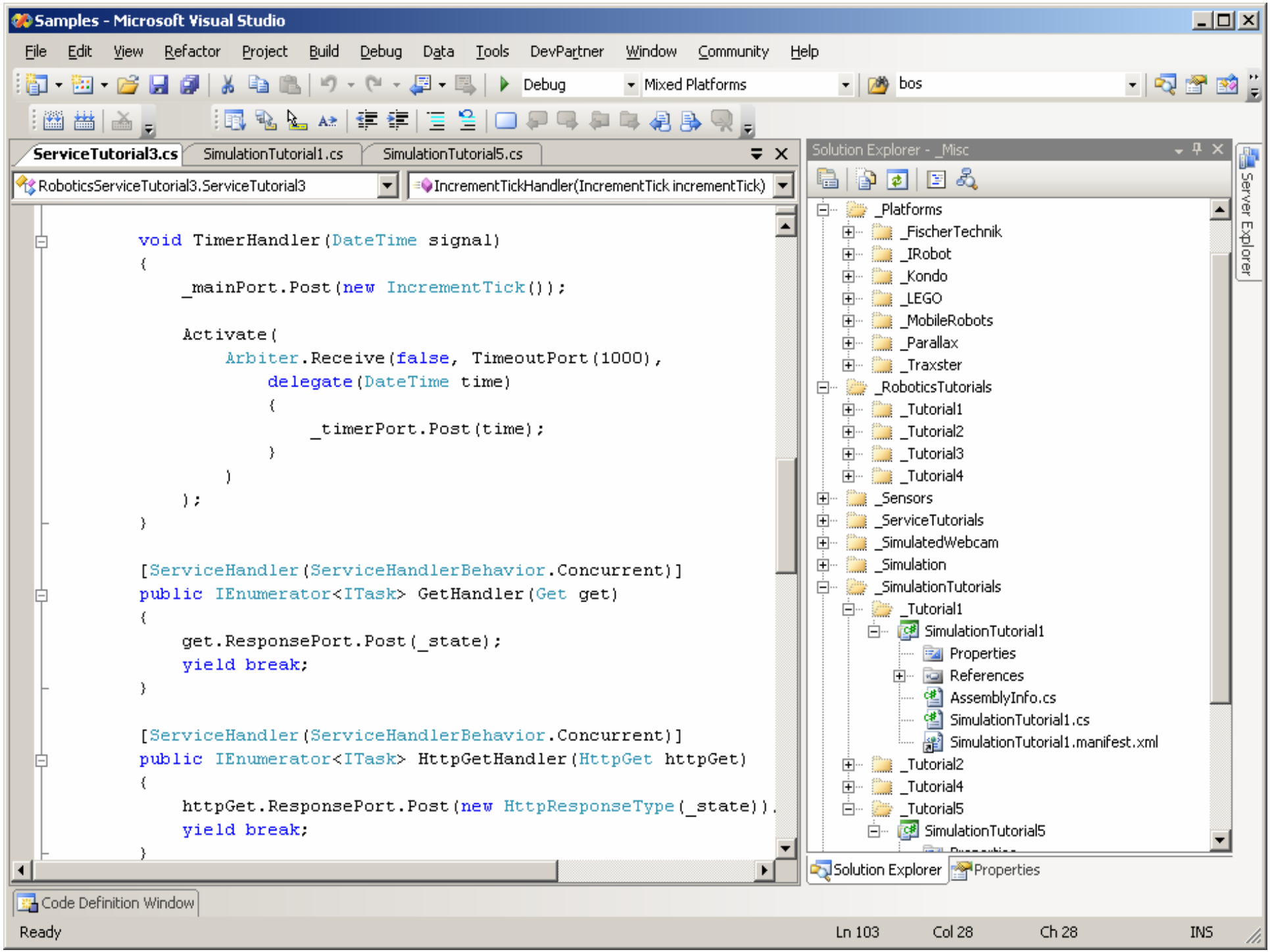
- Dataflow editing
- Model and generate
- Novice to expert

Visual Stuido and .NET tools

- C#
- C++/CLI
- Iron Python
- Visual Basic


http://localhost:50000/simulatedlrf/5231b5bd-ea07-424e-ab38-88a000c74933/sicklrf - Microsoft I...
Datei Bearbeiten Ansicht Favoriten Extras ?
Zurück Suchen
Adresse http://localhost:50000/simulatedlrf/5231b5bd-
<?xml version="1.0" encoding="utf-8" ?>
<State xmlns:s="http://www.w3.org/
xmlns:wsa="http://schemas.xmlsoa
xmlns:d="http://schemas.microsoft
xmlns="http://schemas.microsoft.c
<DistanceMeasurements>
<int>8000</int>
http://localhost:50000/simulatedlrf/5231b5bd-ea07-424e-ab38-88a000c74933/sicklrf - Microsoft I...
Datei Bearbeiten Ansicht Favoriten Extras ?
Zurück Suchen Favoriten
Adresse http://localhost:50000/simulatedlrf/5231b5bd-ea07-424e-ab38- Wechseln zu Links SnagIt
<int>1447</int>
<int>1421</int>
<int>1395</int>
<int>1371</int>
<int>1347</int>
<int>1325</int>
<int>1303</int>
<int>1282</int>
<int>1261</int>
<int>1242</int>
<int>1223</int>
<int>1205</int>
<int>1197</int>
<int>1203</int>
<int>1210</int>
<int>8000</int>
<int>8000</int>
<int>8000</int>
<int>8000</int>
</DistanceMeasurements>
<AngularRange>180</AngularRange>
<AngularResolution>0.5</AngularResolution>
<Units>Millimeters</Units>
<TimeStamp>2008-03-28T22:09:51.0625+01:00</TimeStamp>
<LinkState>Measurement received</LinkState>
</State>
Lokales Intranet


lab2* - Microsoft Visual Programming Language
File Edit View Build Run Help
Basic Activities
Activity
Variable
Calculate
Data
Join
Merge
If
Switch
List
Services
Find service ...
All Found
Announce
Arcos Bumper
Arcos Core
Arcos Drive
Atom Syndication Generator
Atom Syndication Generator 1
Blob Tracker
Blob Tracker Calibrate
Boe-Bot BASIC Stamp 2
Boe-Bot Generic Contact Sens
Boe-Bot Generic Drive
Boe-Bot Generic Motor
ColorSegment
Common DSS Test Implemen
CppRoboticsTutorial1
CppRoboticsTutorial2
CppRoboticsTutorial4
Desktop Joystick
Direction Dialog
Direction Dialog (VB)
DME5000 Szenario
DME5000Visualizer
Explorer
fischertechnik®
Diagram
Timer
FireTimer
Calculate
!state.TurnRight
Calculate
1.1 * state.TurningRadius
Variable
Set
TurnRight
bool
Join
TurnRight
TurningRadius
Variable
Set
TurningRadius
double
CalculateWheelPowers
TurningRadiusToWheelPowers
SetDrivePow
Merge
Data
true
bool
Variable
Set
TurnRight
bool
Join
TurnRight
TurningRadius
Data
1.0
double
Variable
Set
TurningRadius
double
Project
Diagrams
Diagram
Configurations
IRobot.Create.Simulation.Manifest
iRobot.Drive.Manifest.xml
Properties
Comment:
Name:
TurningRadiusToWheelPowers
Configuration:
None
Loaded


Samples - Microsoft Visual Studio
File Edit View Refactor Project Build Debug Data Tools DevPartner Window Community Help
Debug
Mixed Platforms
bos
ServiceTutorial3.cs
SimulationTutorial1.cs
SimulationTutorial5.cs
RoboticsServiceTutorial3.ServiceTutorial3
IncrementTickHandler(IncrementTick incrementTick)
void TimerHandler(DateTime signal)
{
_mainPort.Post(new IncrementTick());
Activate(
Arbiter.Receive(false, TimeoutPort(1000),
delegate(DateTime time)
{
_timerPort.Post(time);
}
)
);
}
[ServiceHandler(ServiceHandlerBehavior.Concurrent)]
public IEnumerator<ITask> GetHandler(Get get)
{
get.ResponsePort.Post(_state);
yield break;
}
[ServiceHandler(ServiceHandlerBehavior.Concurrent)]
public IEnumerator<ITask> HttpGetHandler(HttpGet httpGet)
{
httpGet.ResponsePort.Post(new HttpResponseType(_state));
yield break;
}
Solution Explorer - _Misc
_Platforms
_FischerTechnik
_IRobot
_Kondo
_LEGO
_MobileRobots
_Parallax
_Traxster
_RoboticsTutorials
_Tutorial1
_Tutorial2
_Tutorial3
_Tutorial4
_Sensors
_ServiceTutorials
_SimulatedWebcam
_Simulation
_SimulationTutorials
_Tutorial1
SimulationTutorial1
Properties
References
AssemblyInfo.cs
SimulationTutorial1.cs
SimulationTutorial1.manifest.xml
_Tutorial2
_Tutorial4
_Tutorial5
SimulationTutorial5
Server Explorer
Solution Explorer
Properties
Code Definition Window
Ready
Ln 103
Col 28
Ch 28
INS

```
from Microsoft.Ccr.Core import *
from Microsoft.Dss.Hosting import *
import Microsoft.Dss.ServiceModel.Dssp as dssp
import W3C.Soap as w3c
import Microsoft.Robotics.Services.ContactSensor.Proxy as bumper

manifestLocation = System.IO.Path.Combine(System.Environment.CurrentDirectory,"..\samples\config\LEGO.NXT.MotorTou
DssEnvironment.Initialize(50000, 50001, manifestLocation)

def Shutdown():
  global DssEnvironment
  DssEnvironment.Shutdown()
  System.Environment.Exit(0)

def DirectoryQueryFailure(failure):
  print "Could not find service"
  Shutdown()

def bumperUpdate(notification):
  if notification.Body.Pressed :
    print "Ouch - the bumper was pressed."

def DirectoryQuerySuccess(serviceInfo):
  global DssEnvironment
  try :
    _bumperPort = DssEnvironment.ServiceForwarder[bumper.ContactSensorArrayOperations](System.Uri(serviceInfo.Serv
    bumperNotificationPort = bumper.ContactSensorArrayOperations()
    _bumperPort.Subscribe(bumperNotificationPort)
    print "Subscribing to bumpers..."
    Arbiter.Activate( DssEnvironment.TaskQueue,Arbiter.Receive[bumper.Update](True, bumperNotificationPort, bumper
  except :
    print "Could not subscribe to bumper:", sys.exc_info()[0]
    Shutdown()

Arbiter.Activate(DssEnvironment.TaskQueue, \
  Arbiter.Choice[dssp.ServiceInfoType,w3c.Fault](DssEnvironment.DirectoryQuery(bumper.Contract.Identifier), \
    DirectoryQuerySuccess,DirectoryQueryFailure ))

print "Wait a few seconds for bumpers or press 'Enter' anytime to exit"
System.Console.ReadLine()
Shutdown()
```

Reasons for simulator

Reasons:

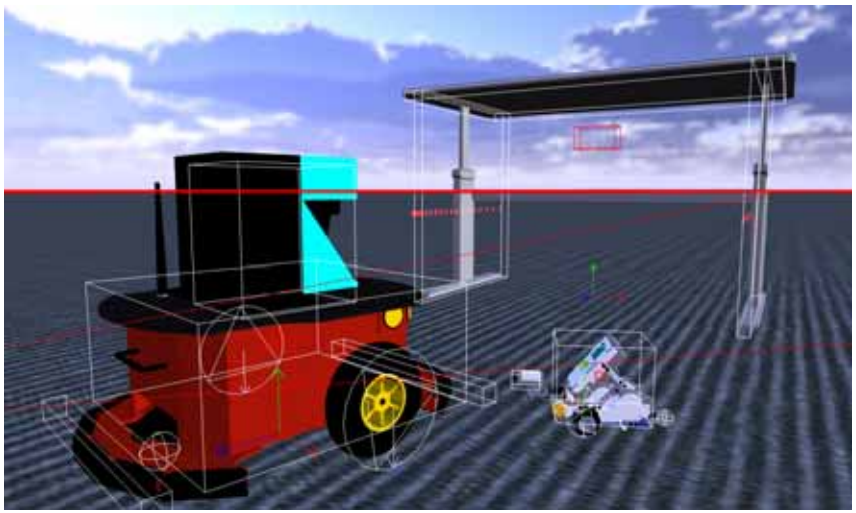
- Robotics hardware is expensive
- Often a limited ressource (team)
- Difficult to debug and reproduce

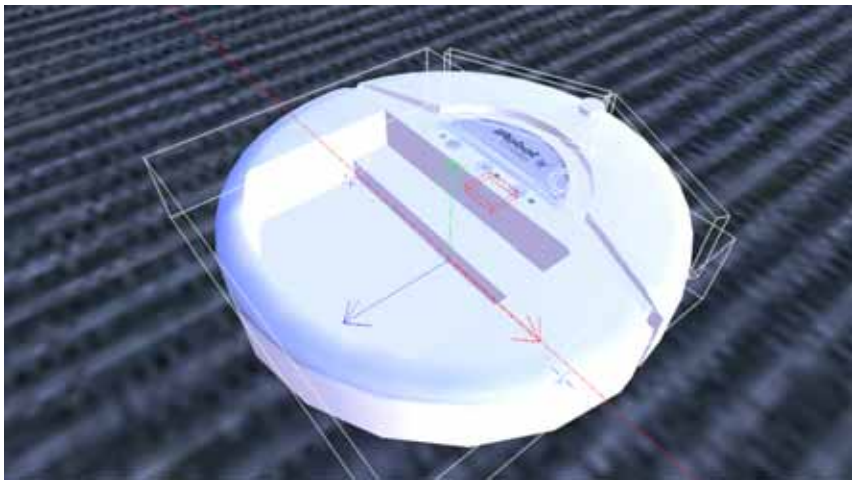
Pros:

- Easy for prototyping
- Low barrier to entry
- Staged approach
- Useful for education

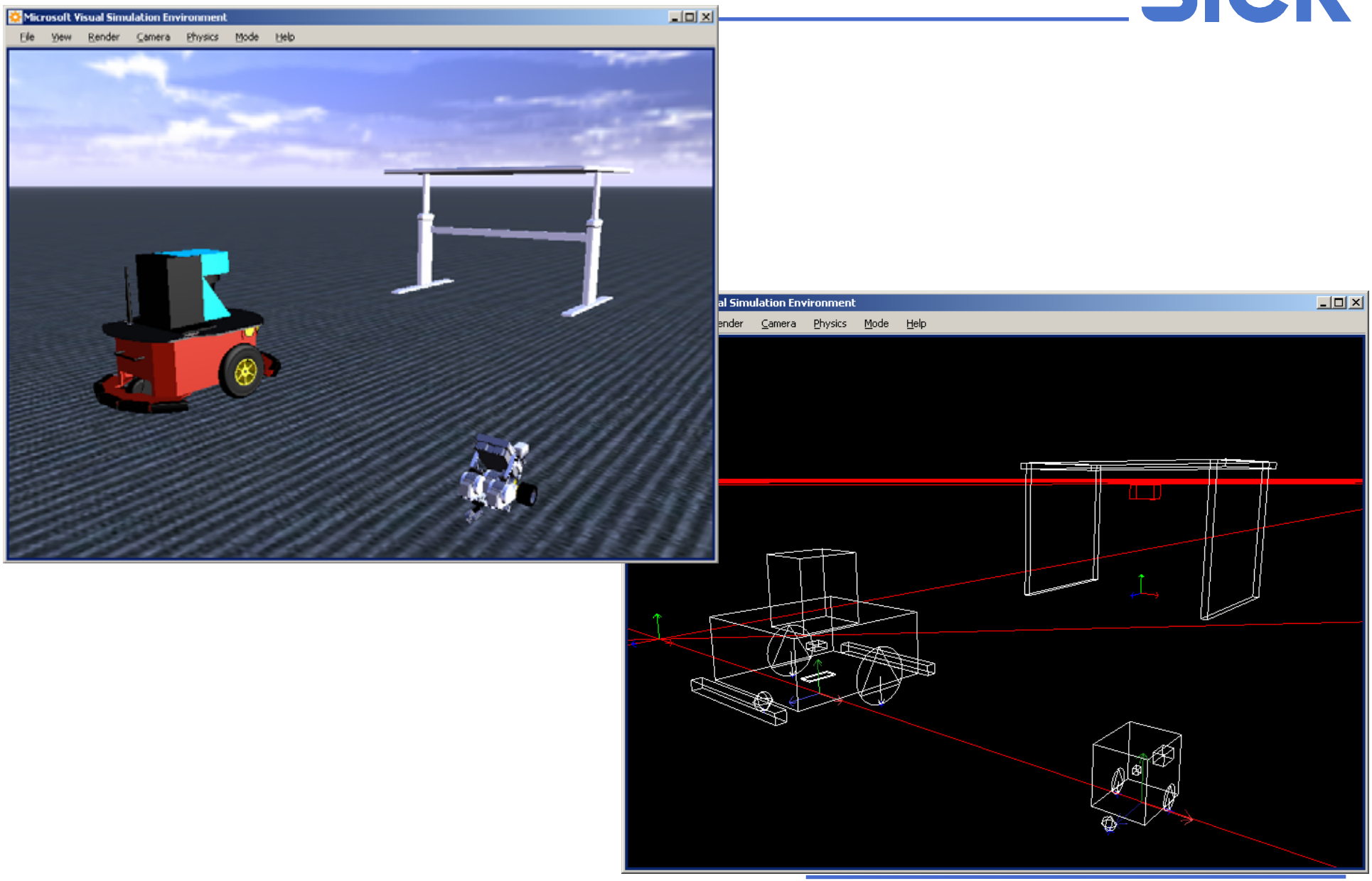
Cons:

- Incomplete model
- Lack of noisy data
- Accurate tuning ?

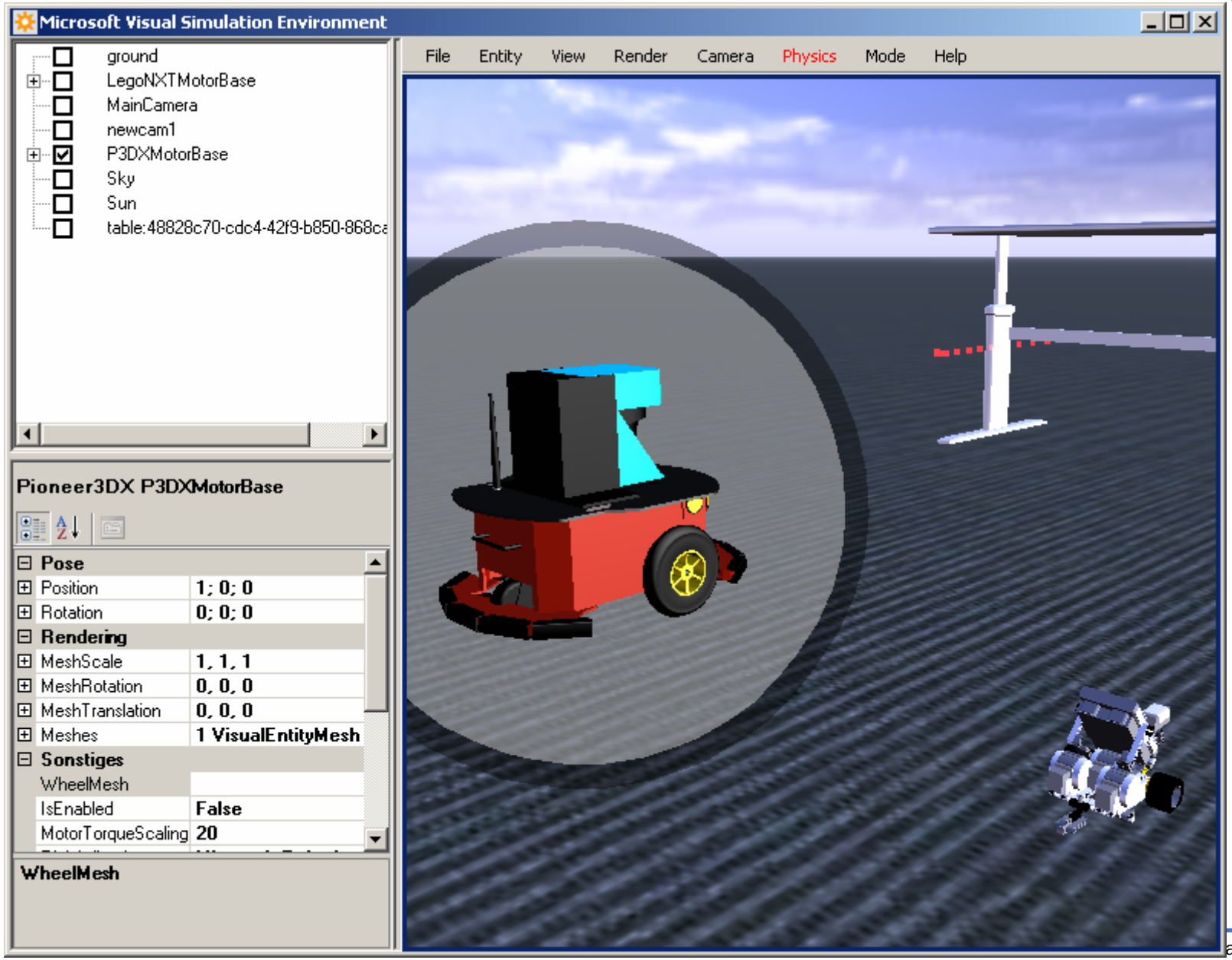




Simulator


Microsoft Visual Simulation Environment
File View Render Camera Physics Mode Help
al Simulation Environment
ender Camera Physics Mode Help

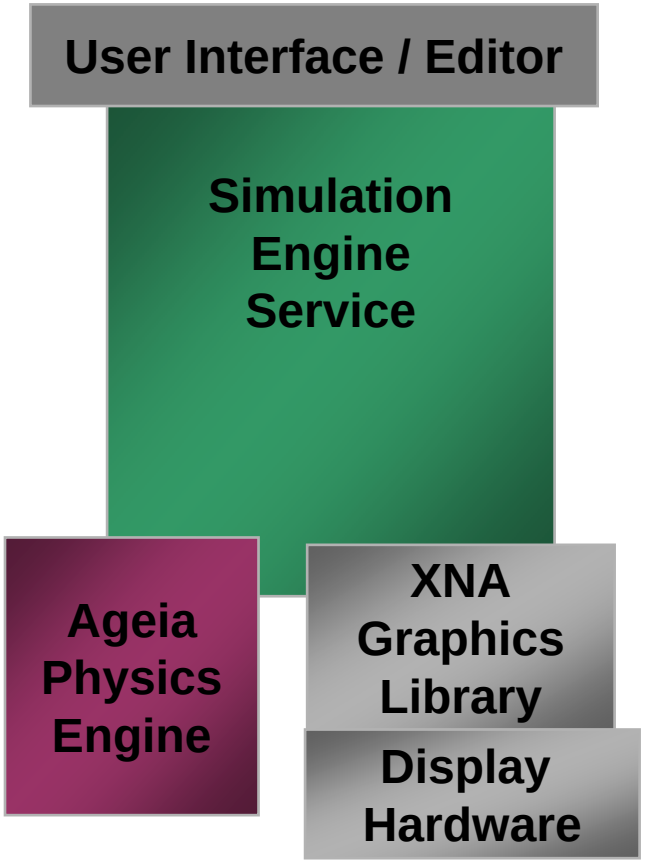
Simulator


Microsoft Visual Simulation Environment
ground
LegoNXTMotorBase
MainCamera
newcam1
P3DXMotorBase
Sky
Sun
table:48828c70-cdc4-42f9-b850-868ca
File
Entity
View
Render
Camera
Physics
Mode
Help
Pioneer3DX P3DXMotorBase
Pose
Position
1; 0; 0
Rotation
0; 0; 0
Rendering
MeshScale
1, 1, 1
MeshRotation
0, 0, 0
MeshTranslation
0, 0, 0
Meshes
1 VisualEntityMesh
Sonstiges
WheelMesh
IsEnabled
False
MotorTorqueScaling
20
WheelMesh

Simulator Architecture: Engine Service

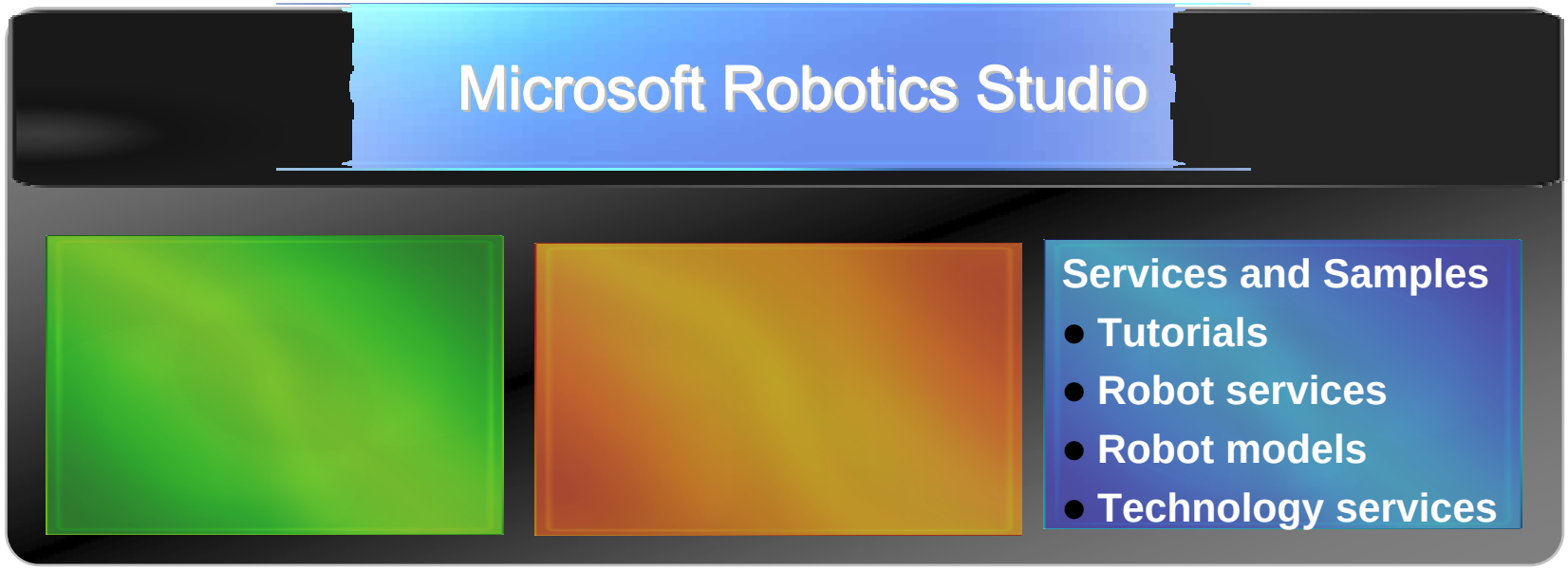
SICK

- Implemented as a service
- Maintains world state
- Manages input devices
- 3D rendering using XNA
- Ageia Physics Simulation
- Graphical User Interface
- Editor for modeling and debugging



Microsoft Robotics Studio

A development platform for the robotics community, supporting a wide variety of users, hardware, and application scenarios

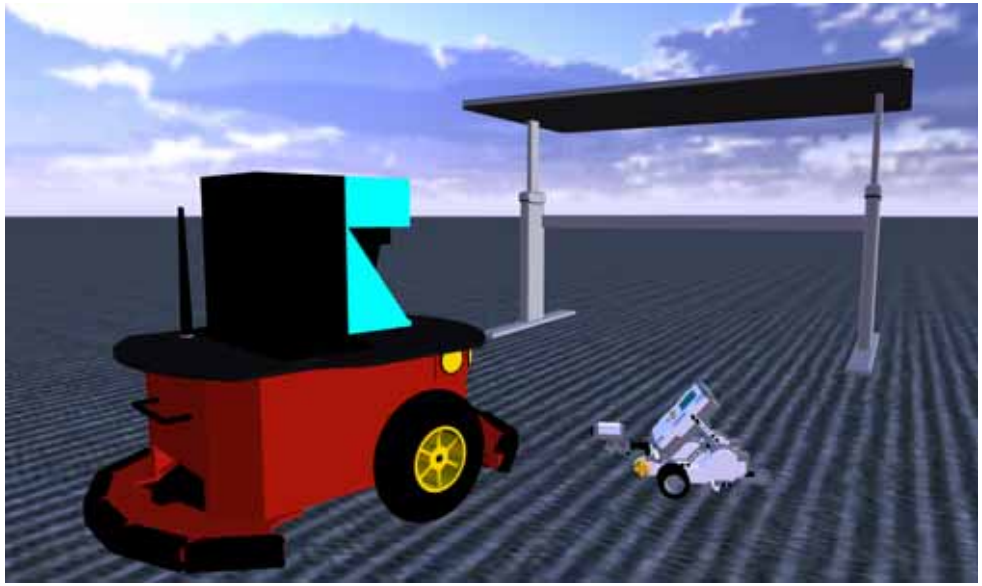


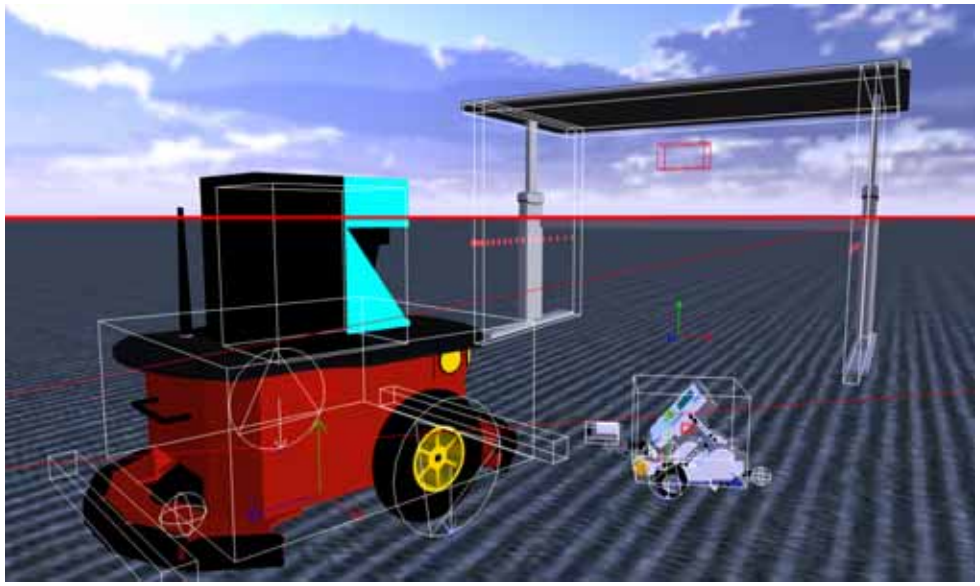
Supported Sensor: SICK LRF

SICK LRF

- Driver in MSRS 1.0/1.5
- Now improved version

Additional on roadmap

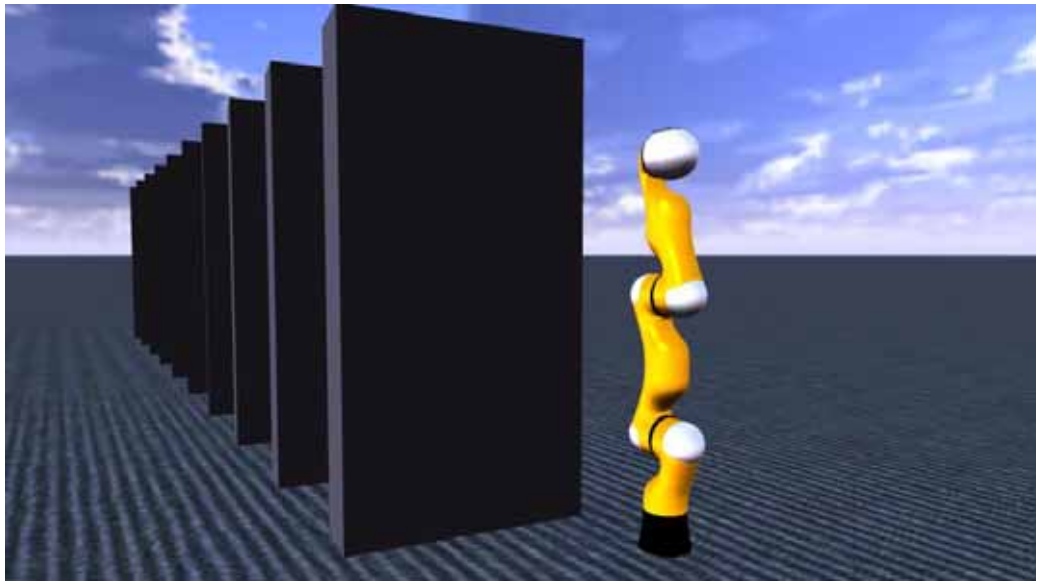


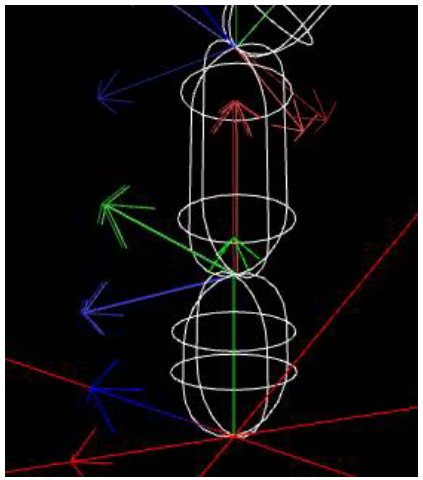


Supported Actuator: KUKA LBR3

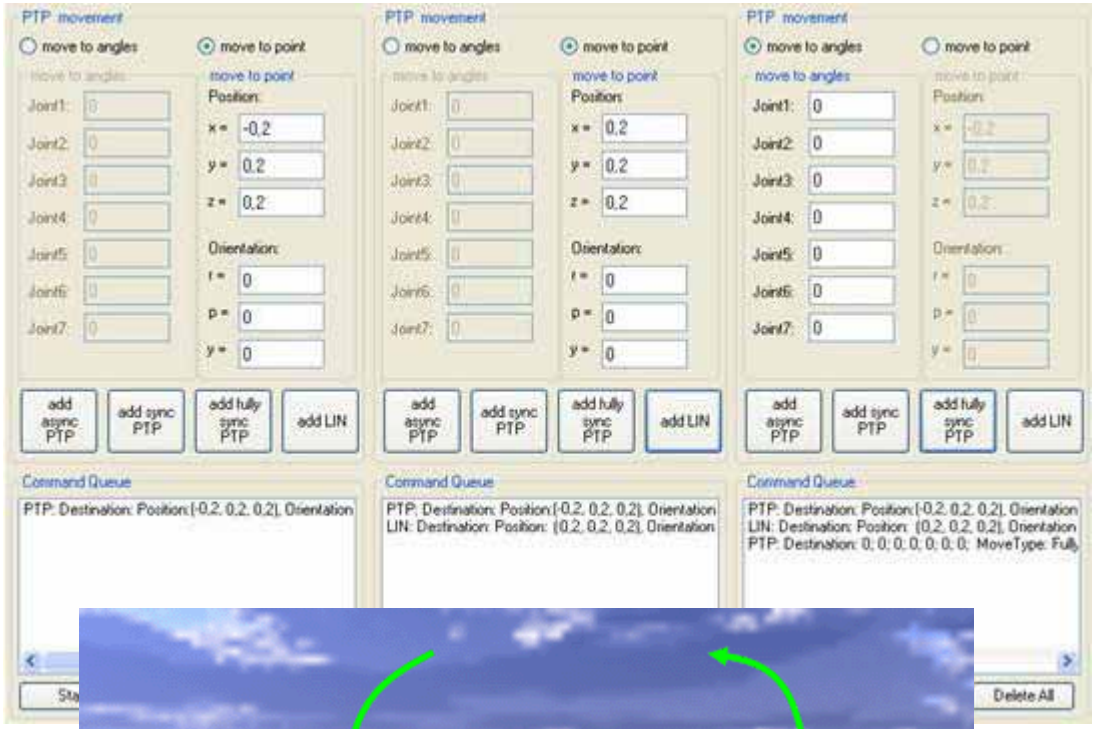
SICK

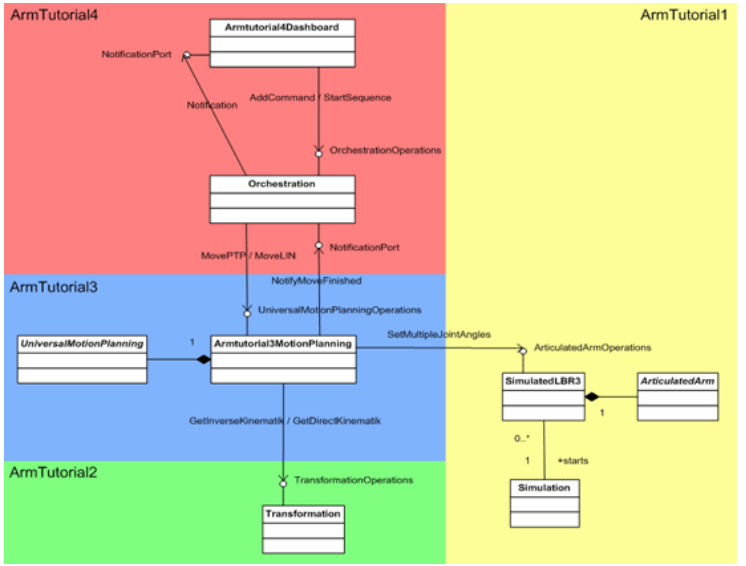
KUKA LBR3

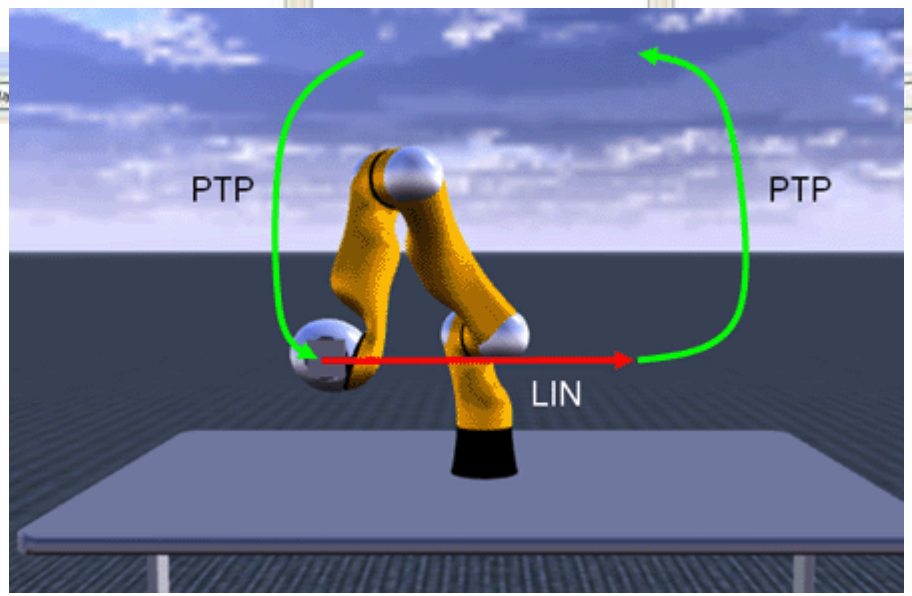


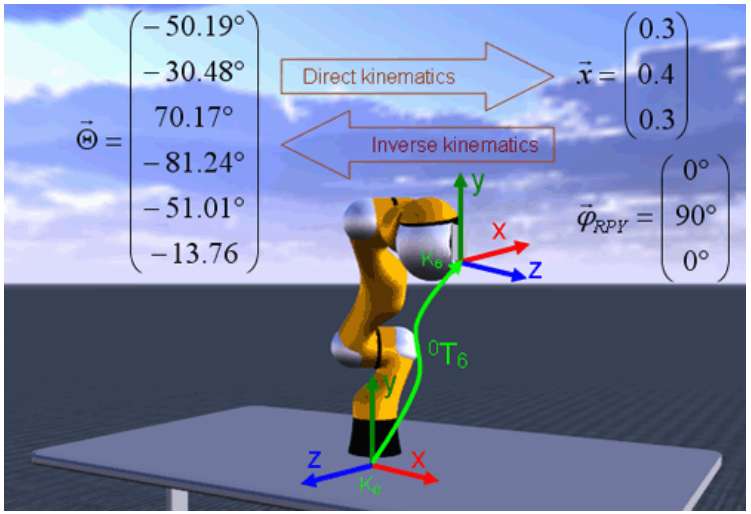


Samples: KUKA Educational Framework

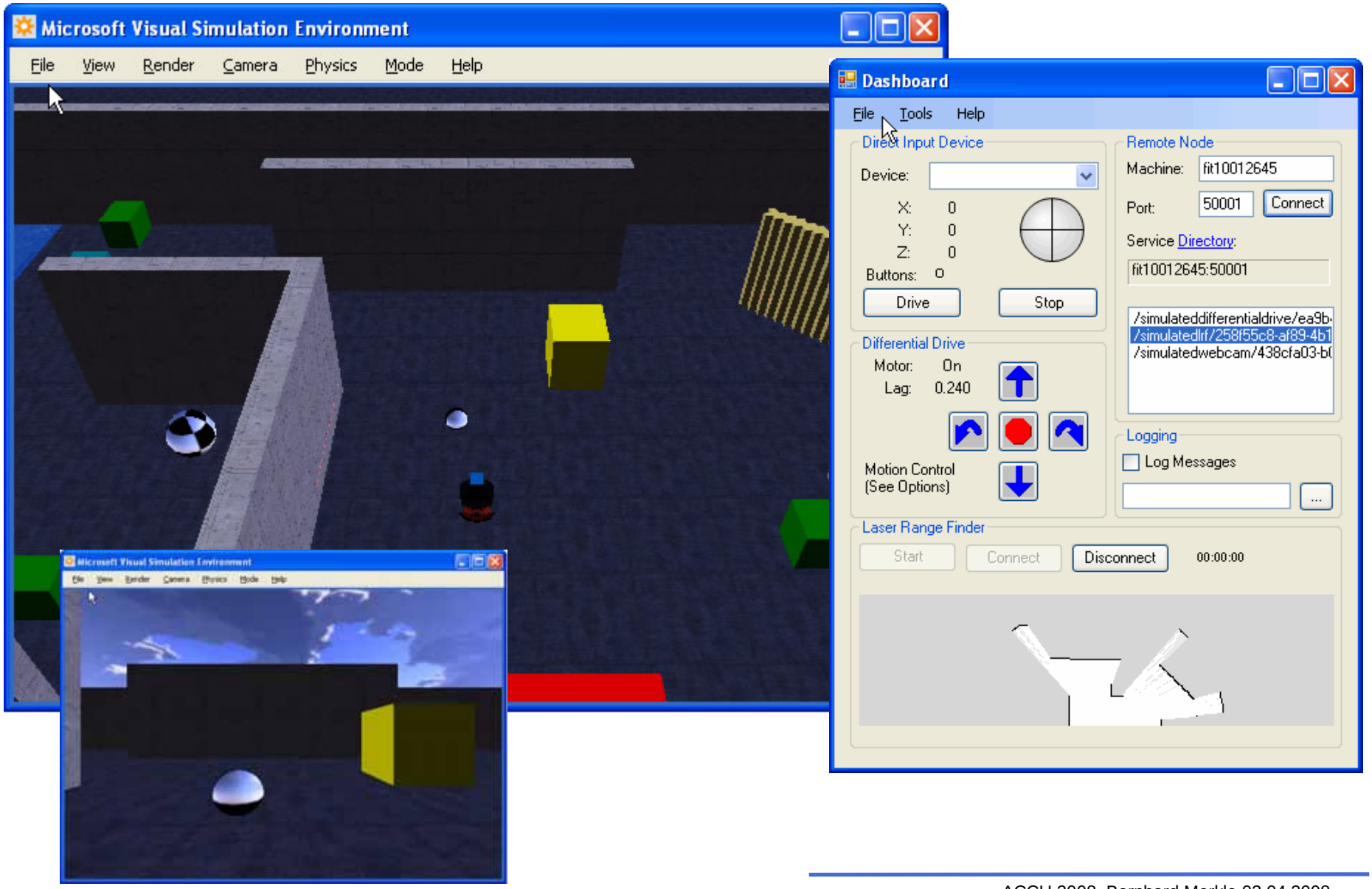




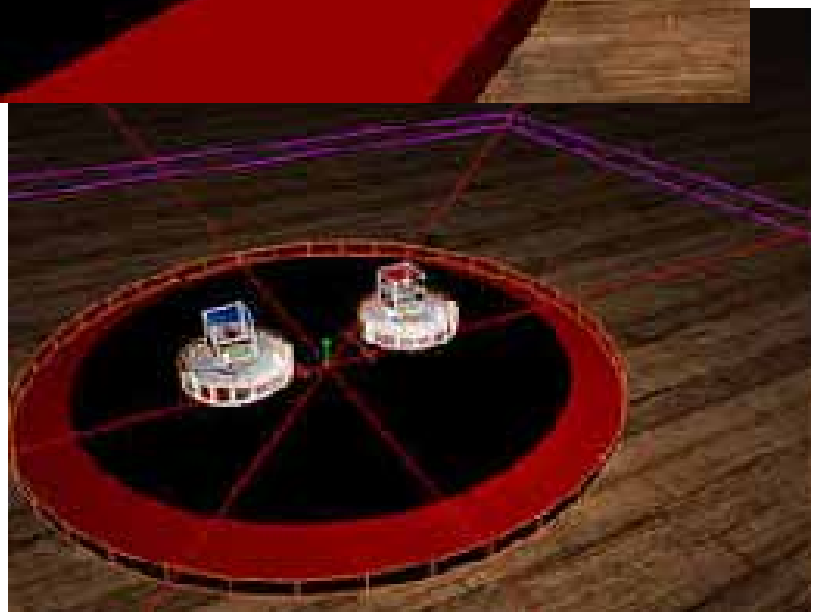




Samples: Maze Simulator



Samples: Sumo



Samples: RoboCup

SICK

MSRS history:

First CTP June 2006,…

MSRS 1.0: December 2006

- First release, include full runtime, parts of authoring, samples

MSRS 1.5: June 2007

- Runtime
  - (DSS,CRR) ported to .NET CF
  - performace improvements
- VPL
  - Code Generation, deployment, Manifest Editor
- Simulator
  - UI, shadows, material editor
- better documentation, new samples,
- Updates/Bugfixes in Aug07+Nov07

Other Robotics Toolkits

SICK

- OpenSource
  - Open Dynamics Engine (ODE)
  - Simbad
  - TeamBots
  - Khepera II
  - Gazebo
  - CARMEN
- Lego Mindstorms and Lego NXT
  - Legos own development tool chain (RIS, NXGen)
  - Extensions: firmware, OS, programming languages (lejos, nqc/nqx, etc)

References

URL:

- Website
  http://www.microsoft.com/robotics
- Newsgroup/Forum
  http://msdn.microsoft.com/robotics/
- Channel9 wiki
  http://channel9.msdn.com/wiki/default.aspx/Channel9.MSRobotics Studio
- “A robot in every home” Bill Gates
  http://go.microsoft.com/?LinkID=5950849
- Parallel programming with .NET
  http://blogs.msdn.com/pfxteam/
- Blog
  http://blogs.msdn.com/msroboticsstudio/default.aspx

References

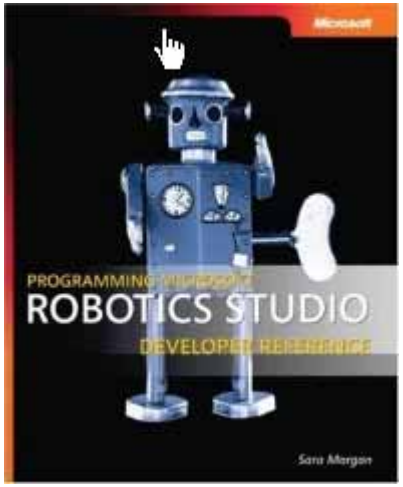
URL:

- Ben Axelrod
  http://www.benaxelrod.com/MSRS/
- Trevor Taylor
  http://sky.fit.qut.edu.au/~taylort2/
- KUKA Educational Framework
  http://www.kuka.com/en/products/software/educational_framework/
- DARPA Urban challenge, PAVE project
  http://pave.princeton.edu/main/
- IBM: Open source robotic toolkits, alphaworks
  http://www.ibm.com/developerworks/linux/library/l-robotools/

References

Books (in summer 2008)

- Programming Microsoft Robotics Studio,
  Sara Morgan
  Microsoft Press



- Professional Microsoft Robotics Studio
  Kyle Johns, Trevor Taylor, (Martin Calsyn)
  Wrox