
MathWorks
Service-Oriented Architectures with Simulink

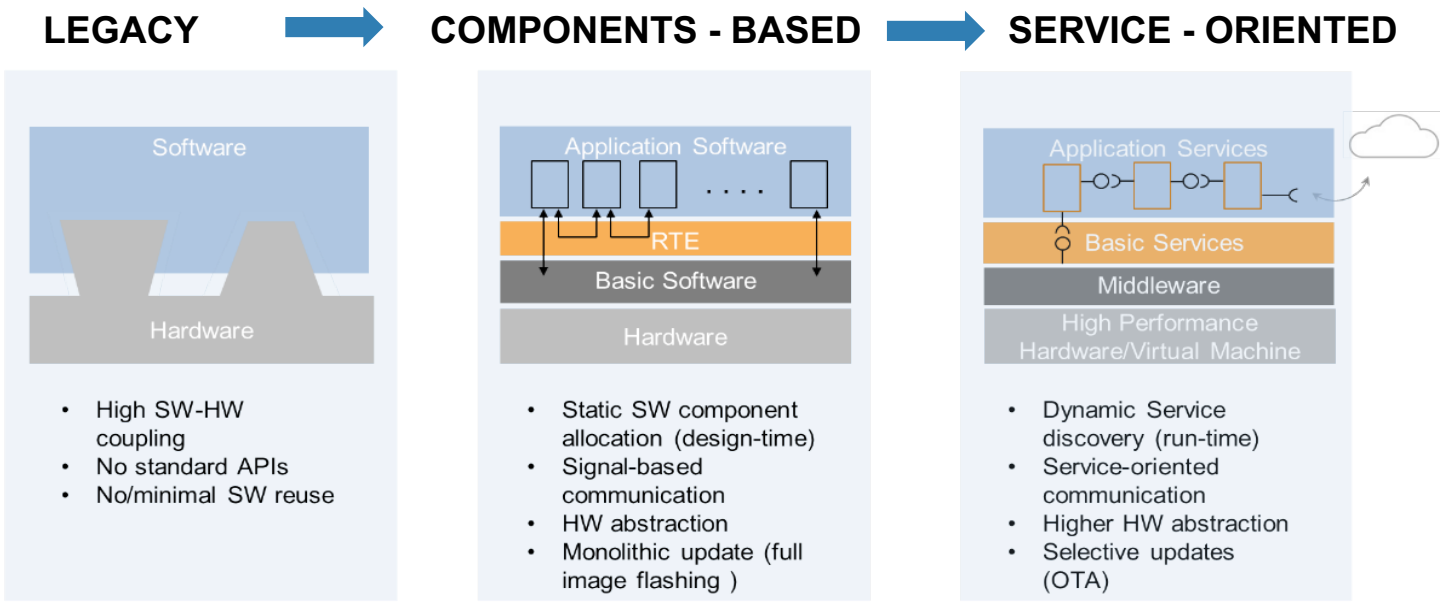
What Is Service-Oriented Architecture?

Service-oriented architecture (SOA) is a type of software architecture that describes a system that consists of a set of services that communicate over messages. It typically involves multiple applications that use these services across platforms.

SOA provides the flexibility to dynamically add, remove, or update components without impacting the entire software system.

This ebook describes how you can use Simulink to model, simulate, and deploy software based on SOA that runs in different applications.

What Is Service-Oriented Architecture? (Continued)



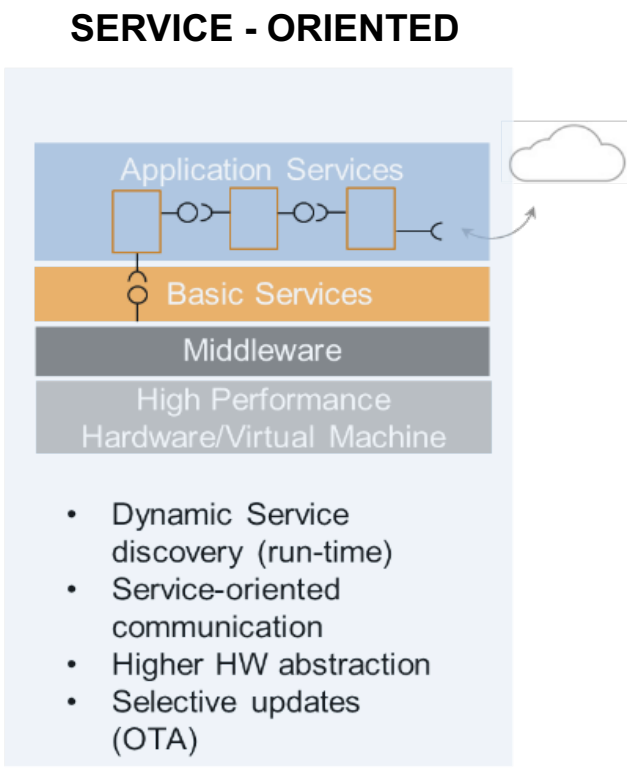
To meet the demand for intelligent applications, such as autonomous driving, modern software architectures are evolving from **legacy**, to **components-based**, to **Service-oriented** architectures.

- Legacy: highly coupled software and hardware with no standard APIs
- Components-based: more standardized, application software is made by components statically allocated to an ECU
- Service-oriented: the applications are developed by composing the services available in the system

What Is Service-Oriented Architecture? (Continued)

SOA provides the architectural framework to integrate applications as independent services that can be easily combined, updated, or upgraded over air.

- SOA based systems become a composition of multiple applications and services that can be distributed on high-performance computers and onto the cloud.
- These systems use service-oriented communication instead of broadcasting signals across the network.
- Information is exchanged among service providers and consumers who subscribe to those providers (more optimized bandwidth usage).



Industry Standards for Service-Oriented Architecture

AUTOSAR Adaptive Platform: The AUTOSAR organization developed this platform based on SOA. The Adaptive Platform provides flexibility and scalability in processing distribution and compute resource allocations required for autonomous driving applications. This helps to update and upgrade adaptive ECU software securely even after its release.

DDS: Data Distribution Services (DDS) uses SOA methodology, and directly addresses publish and subscribe communications for real-time and embedded systems. DDS is used for real-time data exchange in industries such as aerospace and defense, automotive, robotics etc.

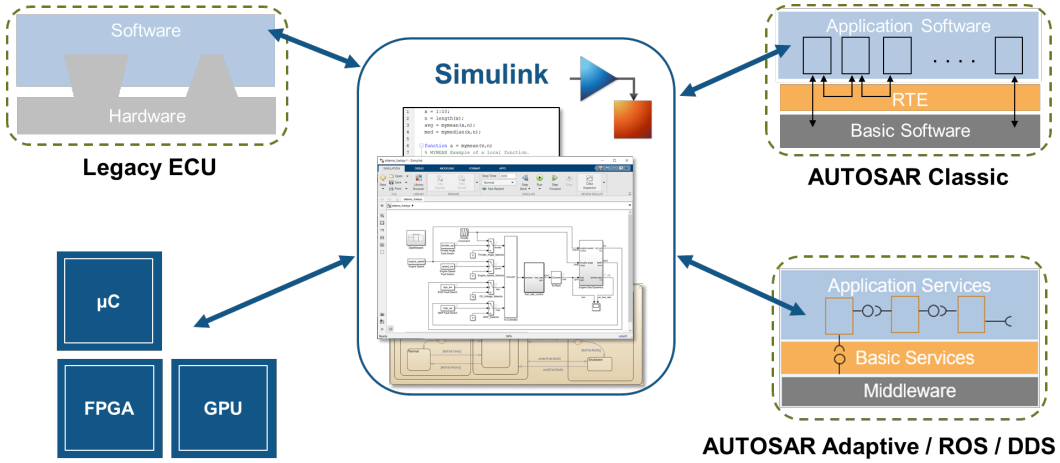
ROS: Many robotics applications use Robot Operating System (ROS), which is based on SOA methodology. ROS serves as a framework for the components necessary to run the software to communicate with each other.

Simulink for Service-Oriented Architecture

Engineers can reuse established workflows and migrate their existing tested components to new architectures.

Using MATLAB and Simulink add-ons you can develop your software once and deploy it to many targets.

You can deploy your models to legacy ECUs, AUTOSAR Adaptive/ROS/DDS applications, or to FPGAs, GPUs and any Linux machines.



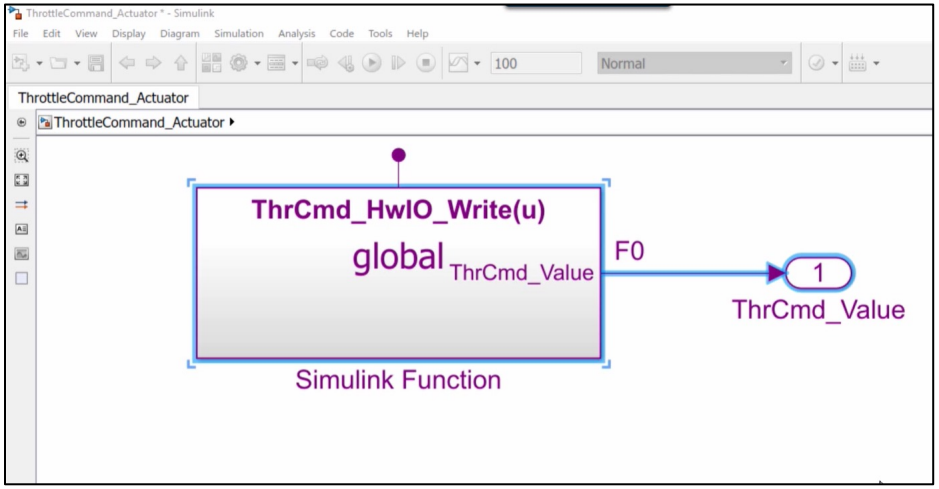
Simulink helps you design service-oriented applications and deploy C++ code compatible with industry standards including AUTOSAR Adaptive, ROS and DDS.

Simulink for Service-Oriented Architecture

Modeling SOA-based services in Simulink

Simulink provides a range of modeling abstractions that help enhance your algorithm model to be suitable for mapping to run-time scheduling and communications based on SOA provided by standard frameworks. For instance, you can use:

- Specific modeling styles to partition your design for run-time scheduling
- Buses to capture software interfaces
- Simulink Functions to model software services
- Schedule Editor and Stateflow® to create simple test harnesses and simulate the composition
- Messages to tie into communication middleware

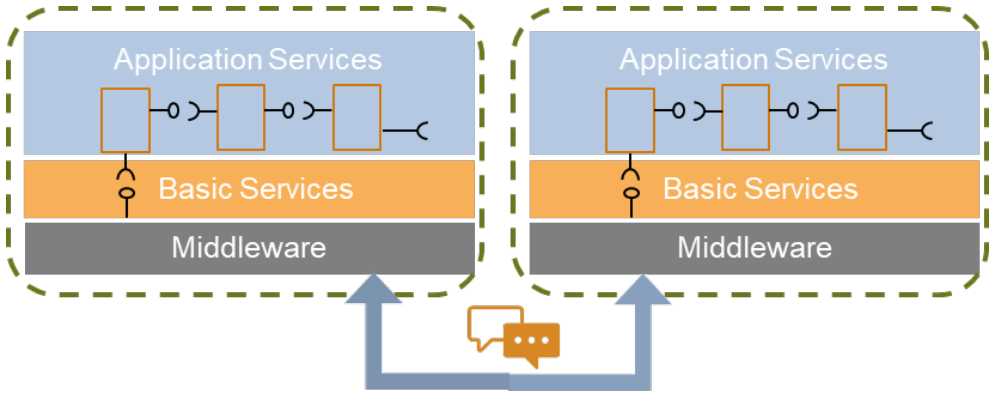


>> How to Model Software Services with Simulink Functions (3:07)

Simulink for Service-Oriented Architecture

Simulating SOA based services in Simulink

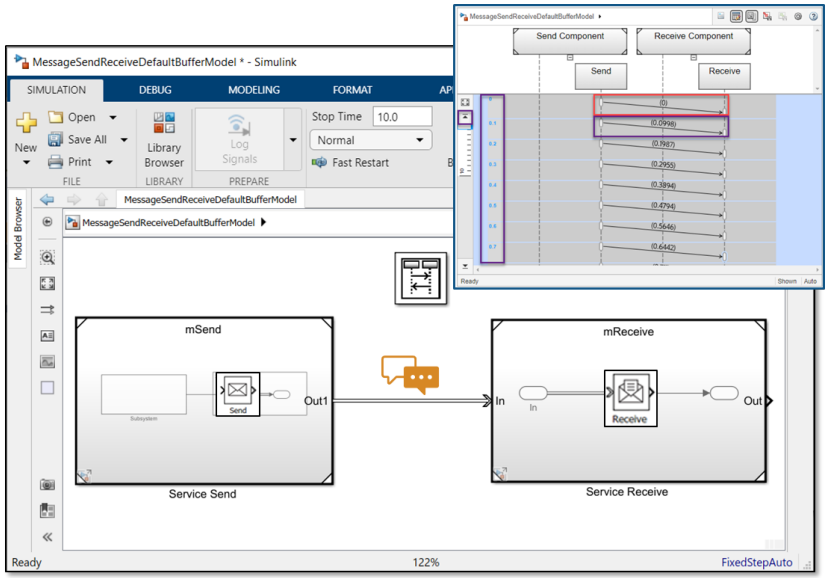
SOA-based application services communicate by sending and receiving messages through a middleware.



Learn how to build a simple model for generating and receiving messages using blocks and debug it using event logging, message animation, and Sequence Viewer.

[>> What Are Messages in Simulink? (2:25)]

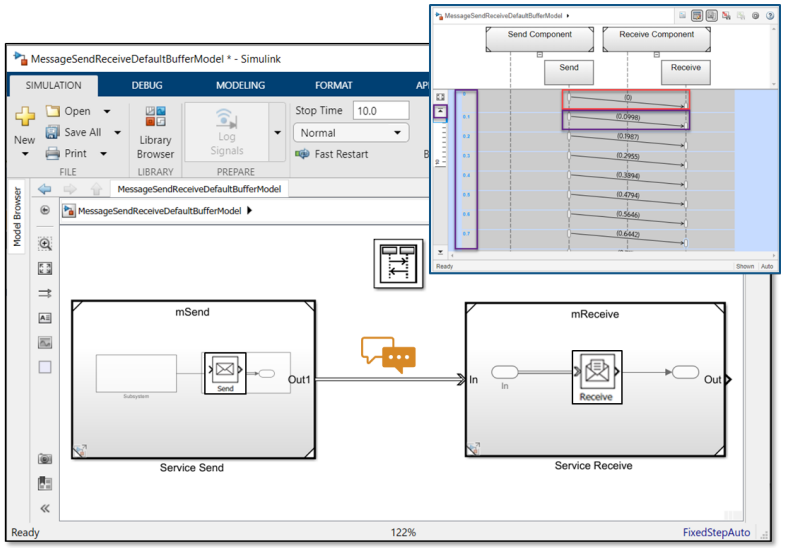
With Simulink messages you can model software compositions with message-based communication. You can send and receive messages from the model's root input and output ports.



Simulink for Service-Oriented Architecture

Deploying SOA using C++

Generate SOA-based C and C++ application code from software services modeled in Simulink for deployment.



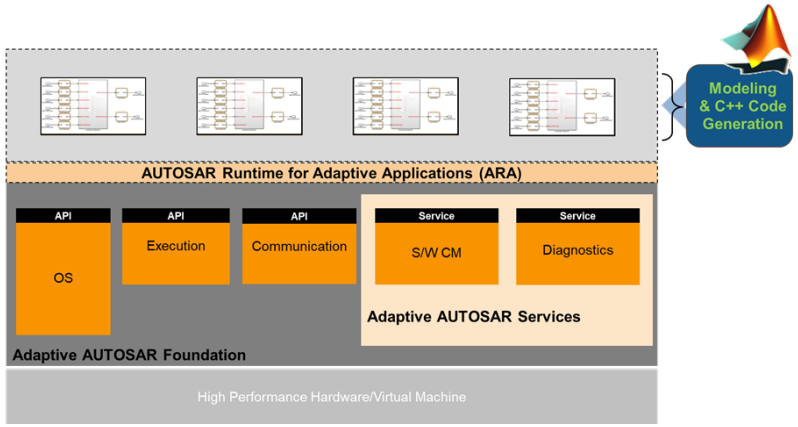

```
// Forward declaration
class MessageSendReceiveDefaultBufferModelModelClass;
class
  MessageSendReceiveDefaultBufferModelModelClassMessa_ReceiveComponent_RecvDataT
  : public RecvData_doubleT
{
 private:
  MessageSendReceiveDefaultBufferModelModelClass & aProvider;
 public:
  MessageSendReceiveDefaultBufferModelModelClassMessa_ReceiveComponent_RecvDataT
    (MessageSendReceiveDefaultBufferModelModelClass & aProvider);
  virtual void RecvData(real_T *data, int32_T *status);
};

class
  MessageSendReceiveDefaultBufferModelModelClassMessa_ReceiveComponent_SendDataT
  : public SendData_doubleT
{
 private:
  MessageSendReceiveDefaultBufferModelModelClass & aProvider;
 public:
  MessageSendReceiveDefaultBufferModelModelClassMessa_ReceiveComponent_SendDataT
    (MessageSendReceiveDefaultBufferModelModelClass & aProvider);
  virtual void SendData(const real_T *data, int32_T *status);
};
```

```
// Constructor
MessageSendReceiveDefaultBufferModelModelClass::
  MessageSendReceiveDefaultBufferModelModelClass():
  ReceiveComponentRecvData(*this)
  ,SendComponentSendData(*this)
  ,Receive_ComponentMDLOBJ0(get_ReceiveComponentRecvData())
  ,Send_ComponentMDLOBJ1(get_SendComponentSendData())
  ,MessageSendReceiveDefaultBuff_M()
{
  // Currently there is no constructor body generated.
}
```

>> Generate C++ Messages to Communicate Between Simulink Components

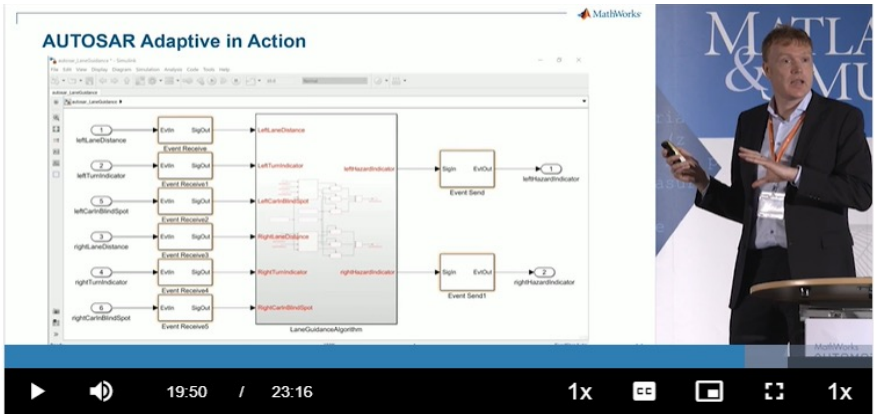
Simulink for AUTOSAR Adaptive

AUTOSAR Adaptive Platform implements the AUTOSAR Runtime for Adaptive Applications (ARA) for automotive industry.



[AUTOSAR Blockset](#) provides apps, blocks and dictionary to model, simulate and generate C++ code for AUTOSAR Adaptive applications in Simulink.

>> Adaptive Software Component Modeling
>> Simulink for Adaptive AUTOSAR (23:16)



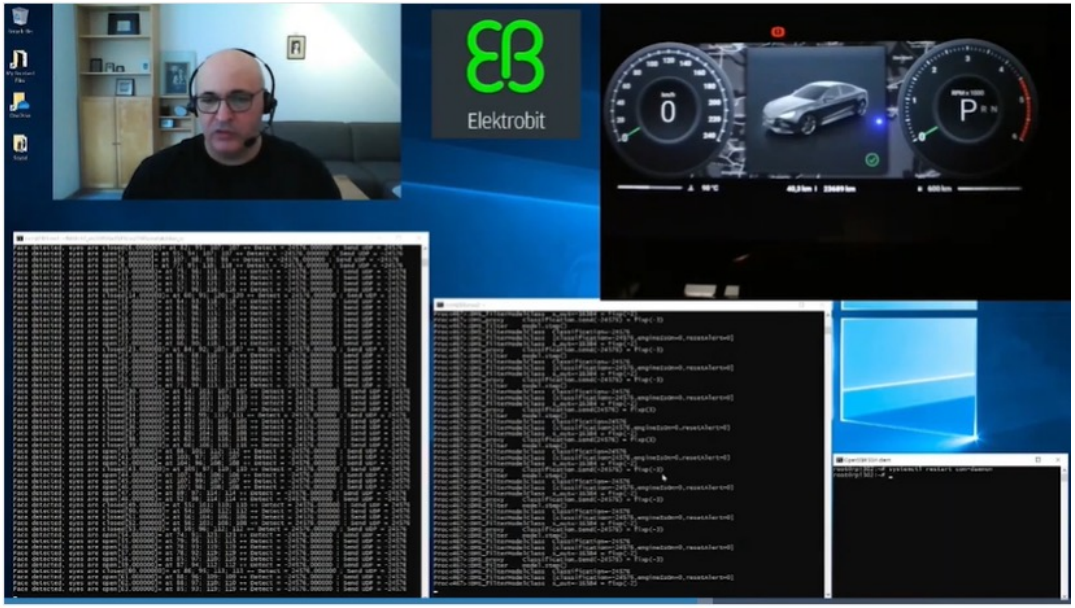
Simulink for AUTOSAR Adaptive

Elektrobit Example

Elektrobit developed a proof-of-concept AUTOSAR Adaptive driver monitoring system using Model-Based Design.

This user example highlights the ease of developing such complex applications using MathWorks tools.

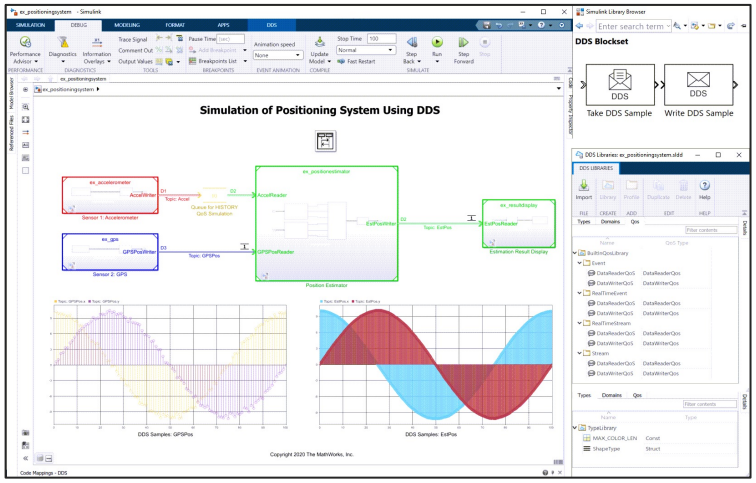
“Developing a Driver Monitoring System with Model-Based Design”
>> Article
>> Presentation (20:50)



Simulink for Data Distribution Services (DDS)

DDS uses a publish-subscribe communication pattern to create a decentralized, architecture-independent, scalable, asynchronous network. It makes it ideal for production-quality data communication in real-time applications.

DDS Blockset provides apps, a dictionary, and blocks for publishing and subscribing samples to DDS, including their corresponding Quality of Service (QoS.). It fully integrates with the RTI Connext DDS and eProsima Fast DDS stacks. The DDS Blockset generates C++ code and XML files from Simulink models.

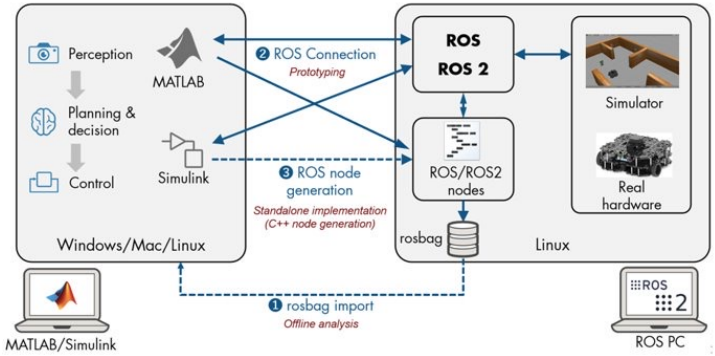


[>> DDS Blockset]

[>> Using DDS Middleware with Simulink (11:32)]

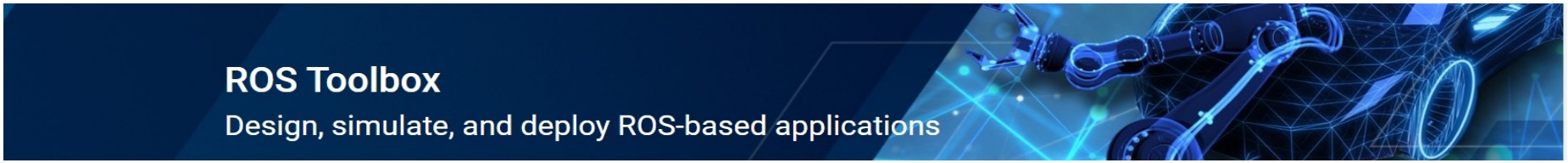
Simulink for Robotics Operating Systems (ROS)

ROS is a communication interface that enables different nodes of a robotics system to discover each other and send and receive data across the ROS network.



ROS Toolbox provides a library of functions that enables you to exchange data with ROS-enabled physical robots or robot simulators such as Gazebo®. It supports C++ code generation to automatically generate ROS nodes from a model and deploy to simulated or real hardware.

[>> ROS Toolbox]
[>> What is ROS Toolbox (2:04)]



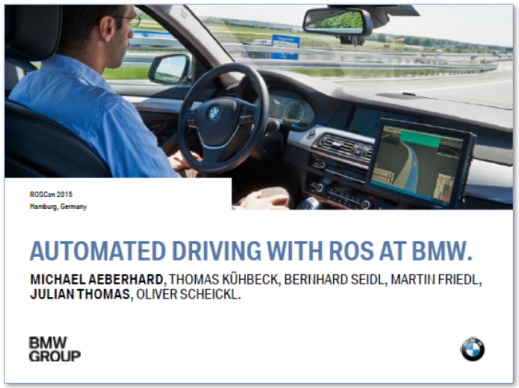
Simulink for Robotics Operating Systems (ROS)

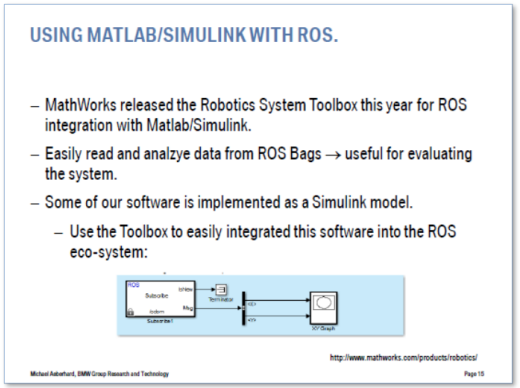
BMW and Voyage Examples

Automated Driving with ROS at BMW

[>> Conference slides]

[>> Video (28:29)]

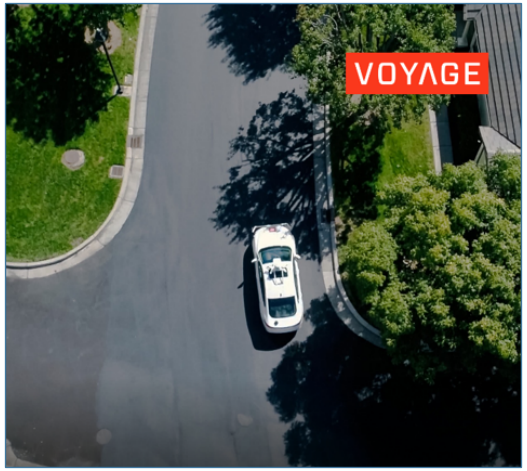




Voyage developed longitudinal controls for a self-driving taxi using Model-Based Design with MATLAB and Simulink. They used ROS Toolbox to build middleware for perception, motion planning, and controls.

As a result, the development speed tripled and easily integrated with open-source software.

[>> Read article]



Learn More

Learn more about using Simulink for SOA

Watch

- Designing and Deploying Service-Oriented Architectures (SOA) with Simulink
- Run Time Software Modeling (Video series)

Read

- What is SOA?

Explore examples

- Model message-based communication between software components
- Getting started with AUTOSAR Adaptive in Simulink
- Getting started with DDS in Simulink
- Getting started with ROS in Simulink