

MATLAB EXPO 2016

Interoperabilität von Simulatoren aus Software Engineering Sicht

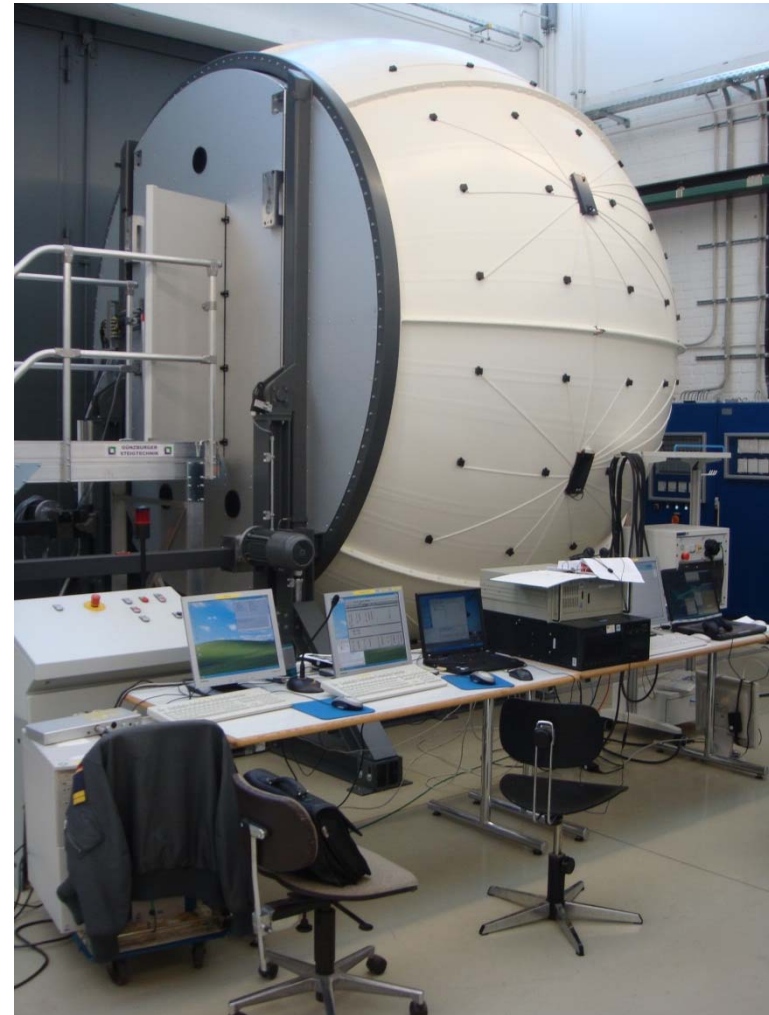
10.05.2016

Prof. Dr.-Ing. R. Finsterwalder
Ingenieurinformatik
Universität der Bundeswehr München

Projekt MASTER:

Entwicklung eines **modularen, re-konfigurierbaren Fahr/Flugsimulators**, der sich sowohl für den Einsatz im Rahmen der **Lehre** als auch für **angewandte Forschung** und **Entwicklung** eignet.

Interoperabilität/Vernetzung von **dislozierten Simulatoren**



- (Flug-)Simulatoren sind komplexe technische Anlagen
- Vielzahl von Komponenten (Dynamiksimulation, Sichtsystem, Daten-IO, Steuerkraftsystem, Audiokommunikation, ...)
- Heterogenität der Komponenten (Software/Betriebssystem-spezifisch, Hardware-spezifisch)
- Komplexität erfordert strikte Modularisierung (Soft- und Hardware)
- Implementierung der Komponenten in Form von unabhängigen und austauschbaren Rechnerprozessen (Modulen)
- Gesamtsimulation = Verteilte Simulation = Summe Einzelsimulationen

- Heterogenität der Komponenten
 - Windows, Linux, QNX, VxWorks
 - PC, Arduino Mikrocontroller, Raspberry PI, B&R X20 CPU
- Echtzeit
 - Schnelle Prozesse (Steuerkraftsystem)
 - Sicherheitskritische Prozesse (Bewegungssystem)
 - Langsame Prozesse (Anzeige Fluginstrumente)
- Komplexität erfordert strikte Modularisierung
 - Interoperabilität der Komponenten erfordert klar definierte Prozessschnittstellen
 - Shared Memory, UDP, TCP, CAN, OSC

- Vielzahl anspruchsvoller (Teil-)Aufgabenstellungen
- Detailliertes Systemwissen und IT-Wissen erforderlich
- Manuelle Low-Level Programmierung ist zeitaufwendig und fehleranfällig
- Applikationsingenieur verfügt über detailliertes Systemwissen, aber ist i.d.R. KEIN IT-Spezialist
- Zahlreiche Entwurfszyklen notwendig (Modell – Regler – Echtzeitcode – Test)
- ➔ Einsatz einschlägiger High-Level Entwicklungswerkzeuge
- ➔ Bereitstellung wiederverwendbarer Bausteine für Prozesskommunikation
- ➔ automatische Codegenerierung

Modul Flugdynamik:

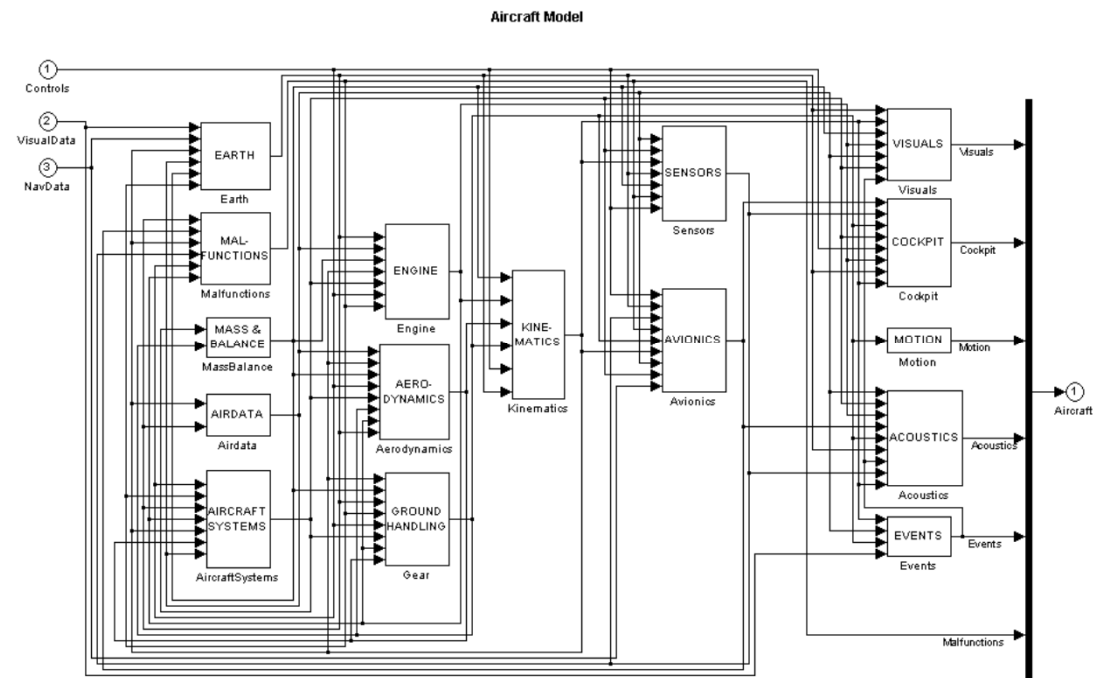
Entwicklung von Simulationsmodellen mit MATLAB/Simulink®

bisher:

- Codierung mit nativen Programmiersprachen (C, C++)

heute:

- problemangepasste grafische oder textuelle Modellbeschreibung mit Modellierungssprache
 ⇒ wiederverwendbare Modelle
- **komfortable Funktionstests**
- **automatische Code-Generierung**
- kurze Entwicklungszeit



Modul Anlagensteuerung:

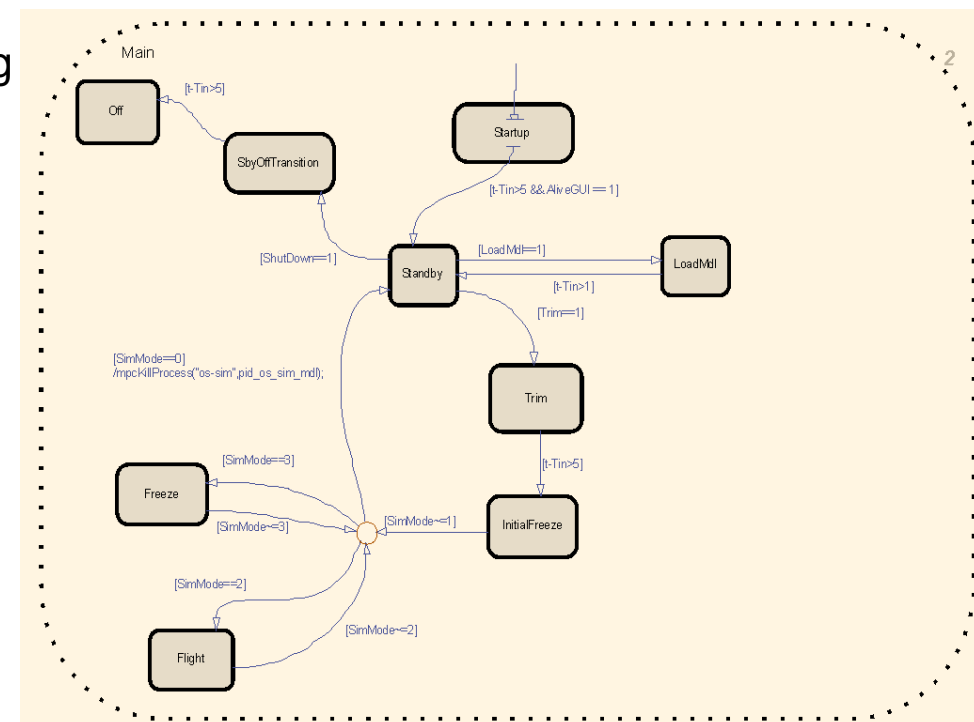
Entwicklung einer Anlagensteuerung der Simulator-Anlage mit MATLAB/Stateflow®

bisher:

- Codierung mit nativer Programmiersprache (C, C++)

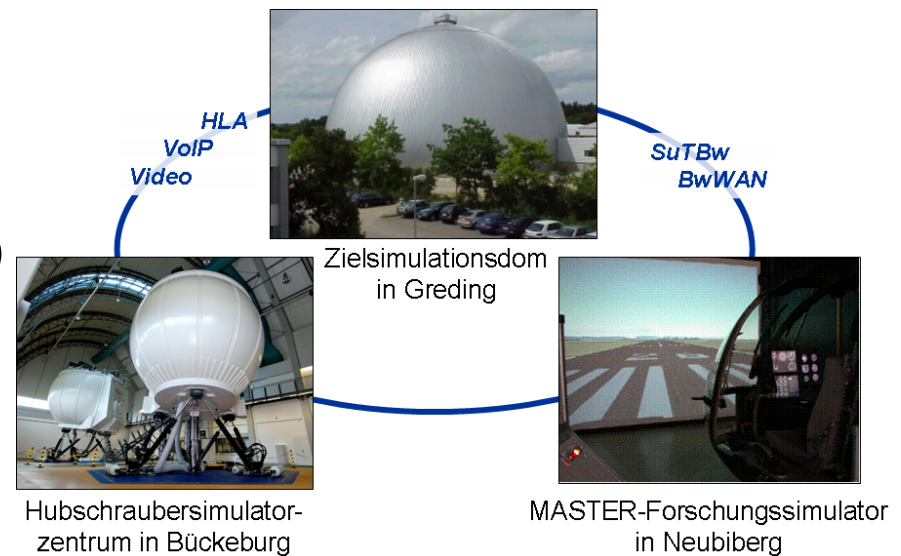
heute:

- Anschauliche grafische Implementierung mit Zustandsdiagrammen
- **einfache Funktionstests**
- **automatische Code-Generierung**
- kurze Entwicklungszeit



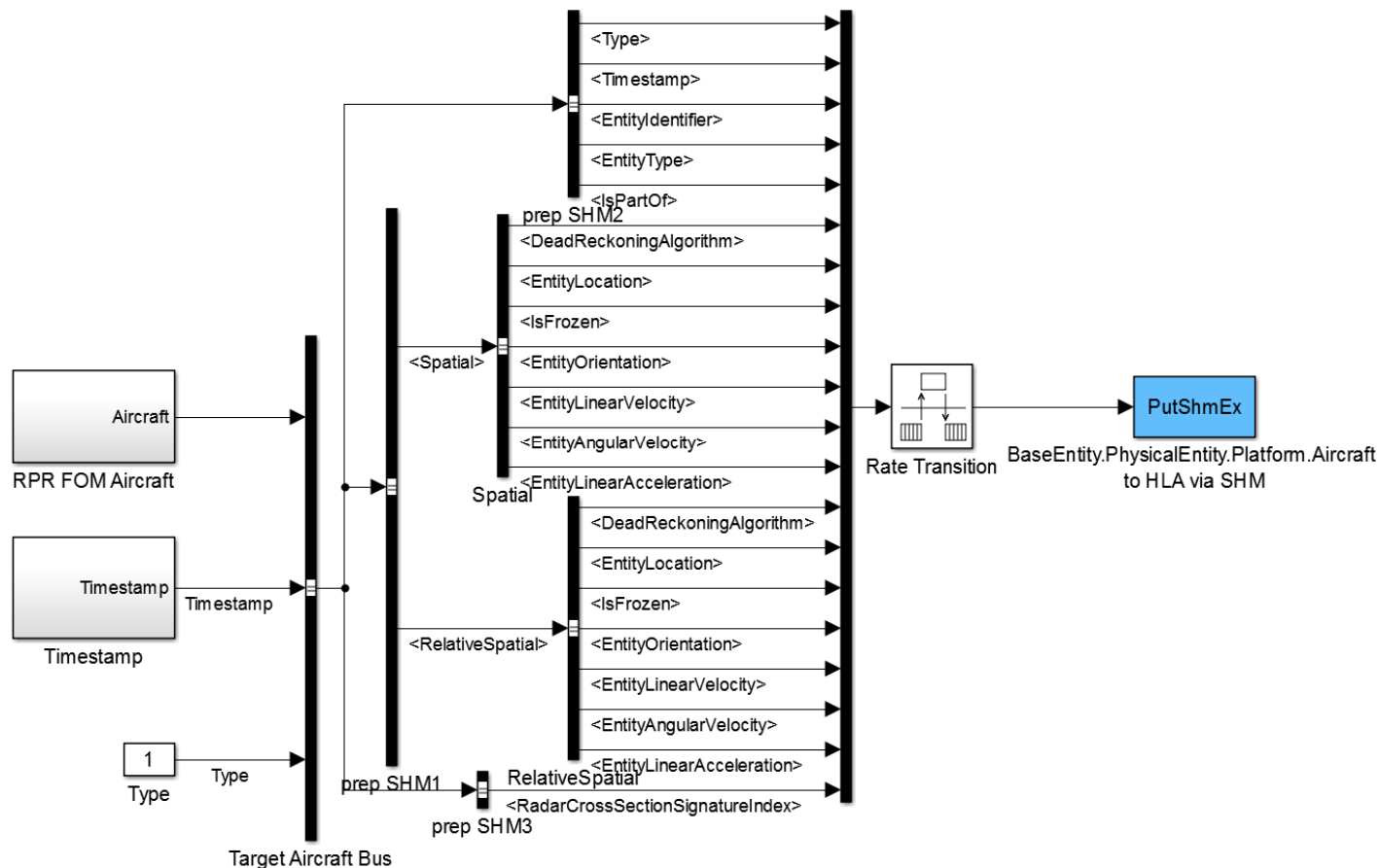
Interoperabilität von Simulatoren

- Training komplexer Szenarien im Verbund
- Austausch von
 - Bewegungsdaten (Fahrzeugdynamik)
 - Audiodaten (Cockpit, Pilot)
 - Audio-/Videodaten (Instructor, Operator)
- Vernetzungsstandards:
 - DIS (udp-Broadcast)
 - HLA (Middleware, RTI, OOP)

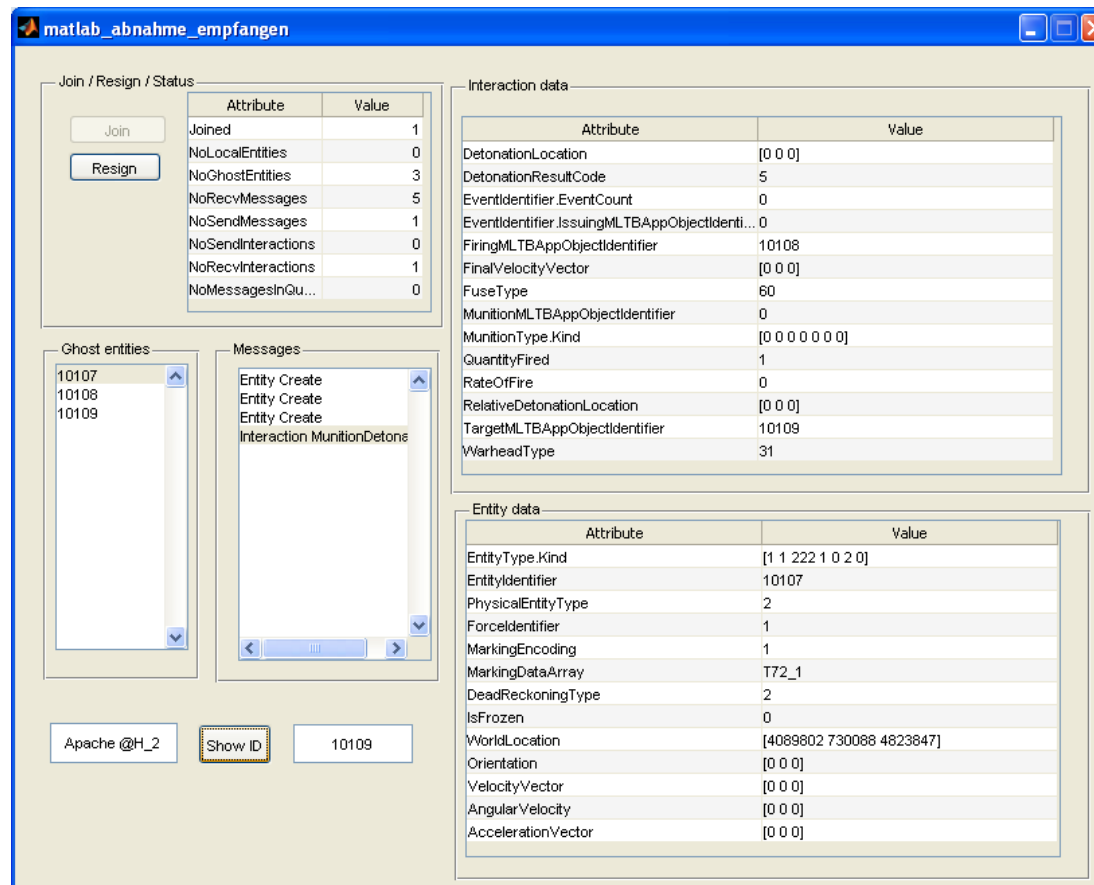


- Vernetzung erfordert detaillierte IT-Kenntnisse
- Applikationsingenieur
 - verfügt über ausgeprägte Applikationskenntnisse
 - ist routiniert im Umgang mit Modellierungs- und Simulationssoftware
 - ist i.d.R. KEIN Netzwerk-Spezialist
- ➔ Entwicklung von wiederverwendbaren Komponenten (S-Functions, MEX-Files)
- ➔ Einfache Instrumentierung eines bestehenden Simulink Simulationsmodells per Drag & Drop für die vernetzte Simulation
- ➔ automatische Codegenerierung von Plattform-spezifischem Echtzeitcode

- Einfache Instrumentierung eines bestehenden Simulink Simulationsmodells per Drag & Drop für die vernetzte Simulation
- Hier: Publizieren der eigenen Bewegungsdaten



- MATLAB m-File Programmierung für portable grafische Benutzeroberflächen
- Hier: Monitoring einer verteilten Simulation



The screenshot shows a MATLAB GUI window titled "matlab_abnahme_empfangen" with several data panels:

- Join / Resign / Status:** Contains "Join" and "Resign" buttons and a table of attributes and values.
- Interaction data:** A table listing various interaction attributes and their values.
- Entity data:** A table listing entity attributes and their values.
- Ghost entities:** A list box containing IDs 10107, 10108, and 10109.
- Messages:** A list box containing messages like "Entity Create" and "Interaction MunitionDetone".
- Bottom controls:** Includes a label "Apache @H_2", a "Show ID" button, and a text field containing "10109".

Attribute	Value
Joined	1
NoLocalEntities	0
NoGhostEntities	3
NoRecvMessages	5
NoSendMessage	1
NoSendInteractions	0
NoRecvInteractions	1
NoMessagesInQu...	0

Attribute	Value
DetonationLocation	[0 0 0]
DetonationResultCode	5
EventIdentifier.EventCount	0
EventIdentifier.IssuingMLTAppObjectIdenti...	0
FiringMLTAppObjectIdentifier	10108
FinalVelocityVector	[0 0 0]
FuseType	60
MunitionMLTAppObjectIdentifier	0
MunitionType.Kind	[0 0 0 0 0 0]
QuantityFired	1
RateOfFire	0
RelativeDetonationLocation	[0 0 0]
TargetMLTAppObjectIdentifier	10109
WarheadType	31

Attribute	Value
EntityType.Kind	[1 1 222 1 0 2 0]
EntityIdentifier	10107
PhysicalEntityType	2
ForceIdentifier	1
MarkingEncoding	1
MarkingDataArray	T72_1
DeadReckoningType	2
IsFrozen	0
WorldLocation	[4089802 730088 4823847]
Orientation	[0 0 0]
VelocityVector	[0 0 0]
AngularVelocity	[0 0 0]
AccelerationVector	[0 0 0]

Unsere Testanlage:

Dislozierung:

- UniBwM / Neubiberg
- Ludwig-Bölkow-Campus / Ottobrunn

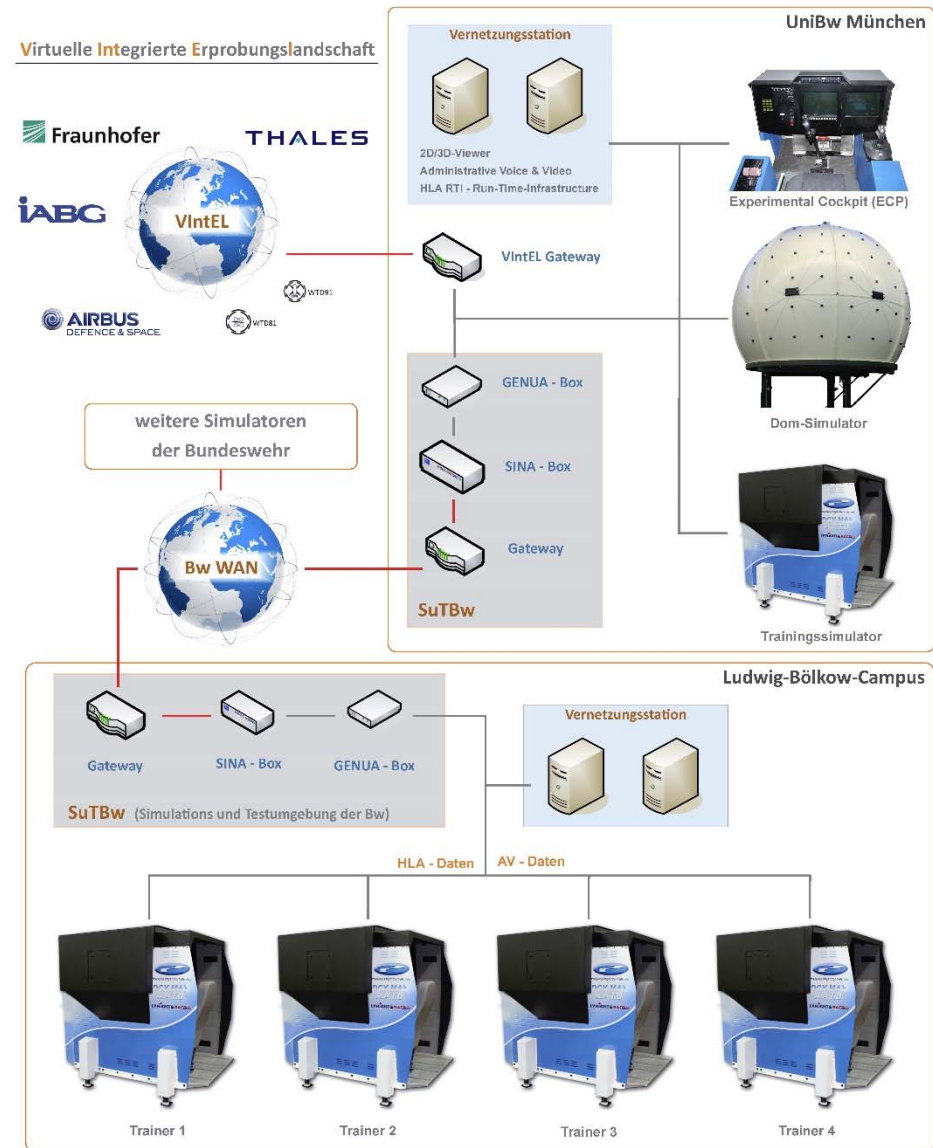
Simulations- und Testumgebung der Bundeswehr (SuTBw):

- Kryptographie-Rechner (Sina Box)
- VPN-Router (Genua)

Virtuelle Integrierte Erprobungslandschaft (VIntEL)

Kooperation mit

- Forschungsinstituten und
- Industrie



Bewertung aus Software Engineering Sicht

- + Interoperabilität – Programmierschnittstellen (MEX, S-Function)
- + Skalierbarkeit – einfache Hardware (Arduino) $\leftarrow \rightarrow$ leistungsfähige Hardware (B&R X20 CPU)
- + Portierbarkeit – automatische Codegenerierung
- + Wiederverwendbarkeit – wiederverwendbare Bausteine
- + Erweiterbarkeit – High-Level-Ansatz
- + Wartbarkeit – Anwendungsspezifische Modellierung

Unsere Erfahrungen mit MATLAB für vernetzte Simulationen:

- Applikation steht im Fokus
 - Bereitstellung wiederverwendbarer Komponenten entlastet Applikationsingenieur
 - Toolkette MATLAB/Simulink, MATLAB/Stateflow - MATLAB/Simulink Coder - Echtzeitcode funktioniert
 - Kurze Zykluszeiten im Entwicklungsprozess
 - Hohe Qualität des erzeugten Programmcodes
 - Unabhängig von der Hardware
- ➔ Modellbasierte Simulation und Entwurf in Verbindung mit automatischer Codegenerierung bieten großes Potential für vernetzte Simulationsanwendungen