Experiences from the application of distributed simulation and their possible relevance for computational systems biology

Steffen Strassburger

-

Ilmenau University of Technology

Ilmenau, Germany

Contents

- **State-of-the-Art in Distributed Simulation**
- Experiences
  - Case Study “Supply Chain Simulation”
  - Case Study “Training Simulation”
  - Case Study “Digital Manufacturing”
  - Case Study “Agent-based Simulation”
- Trends and Potential Impact
- Summary

Distributed Simulation - State-of-the-Art

- Distributed Simulation (DS) – a definition:
  - technology that enables a simulation program to be executed on distributed computer systems [Fujimoto 2000]
- Core algorithms have been around for many years
  - Synchronization: conservative/optimistic/hybrid
  - Efficient Data Distribution: Publish/Subscribe Paradigms, Routing Spaces, Dead Reckoning
- Different motivations for DS: From speed-up to interoperability

Distributed Simulation - State-of-the-Art

- HLA as the leading (but not undisputed) IEEE standard for distributed simulation
  - HLA: High Level Architecture for Modeling and Simulation
  - IEEE 1516-2010 (“HLA Evolved”) as latest version
  - Advantages:
    - Robust industry standard
    - Out of the box solutions for synchronization and efficient data exchange
    - Commercial as well as open source software available
  - Disadvantages:
    - Complex & heavy
    - Limited extensibility

Contents

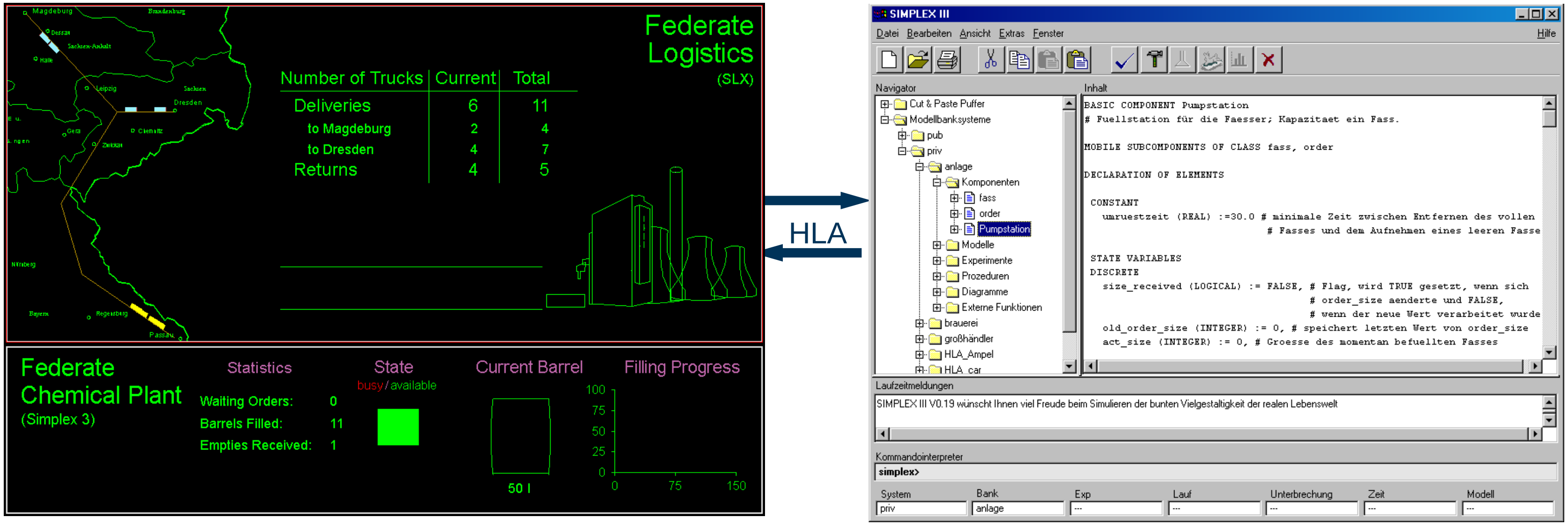
- State-of-the-Art in Distributed Simulation
- **Experiences**
  - Case Study “Supply Chain Simulation”
  - Case Study “Training Simulation”
  - Case Study “Digital Manufacturing”
  - Case Study “Agent-based Simulation”
- Trends and Potential Impact
- Summary

Case Studies - Background

- Involvement with HLA since 1997
- First prototypes for “federating” discrete-event simulation systems (“COTS simulators”) including SLX, Simplex III, Pro Model, Automod, QUEST and IGRIP
- Applications in production and logistics
- From prototypes to industrial applications: Deere & Co. as latest industry partner

Case Study I – Supply Chain Simulation

- Combination of (discrete) logistic model with (continuous) model of chemical production facility



Federate Logistics (SLX)

Number of Trucks	Current	Total
Deliveries	6	11
to Magdeburg	2	4
to Dresden	4	7
Returns	4	5

Federate Chemical Plant (Simplex 3)

Statistics

Waiting Orders: 0
Barrels Filled: 11
Empties Received: 1

State
busy / available
🟩

Current Barrel
50 l

Filling Progress

```
BASIC COMPONENT Pumpstation
# Fuellstation für die Faesser; Kapazitaet ein Fass.

MOBILE SUBCOMPONENTS OF CLASS fass, order

DECLARATION OF ELEMENTS

 CONSTANT
   umruestzeit (REAL) :=30.0 # minimale Zeit zwischen Entfernen des vollen
                             # Fasses und dem Aufnehmen eines leeren Fasse

 STATE VARIABLES
 DISCRETE
   size_received (LOGICAL) := FALSE, # Flag, wird TRUE gesetzt, wenn sich
                                     # order_size aenderte und FALSE,
                                     # wenn der neue Wert verarbeitet wurde
   old_order_size (INTEGER) := 0, # speichert letzten Wert von order_size
   act_size (INTEGER) := 0, # Groesse des momentan befuellten Fasses
```

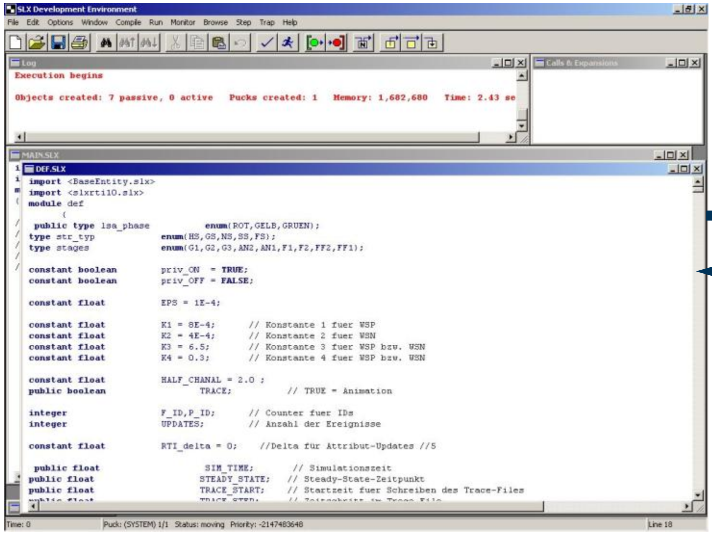
```
SIMPLEX III V0.19 wünscht Ihnen viel Freude beim Simulieren der bunten Vielgestaltigkeit der realen Lebenswelt
```

```
simplex>
```

System	Bank	Exp	Lauf	Unterbrechung	Zeit	Modell
priv	anlage	---	---	---	---	---

Case Study II – Training Simulation

- Combination of a real-time driving simulator with an event-based traffic simulation


SLX Development Environment
File Edit Options Window Compile Run Monitor Browse Step Trap Help
Log
Calls & Expansions
MAIN.SLX
DEF.SLX

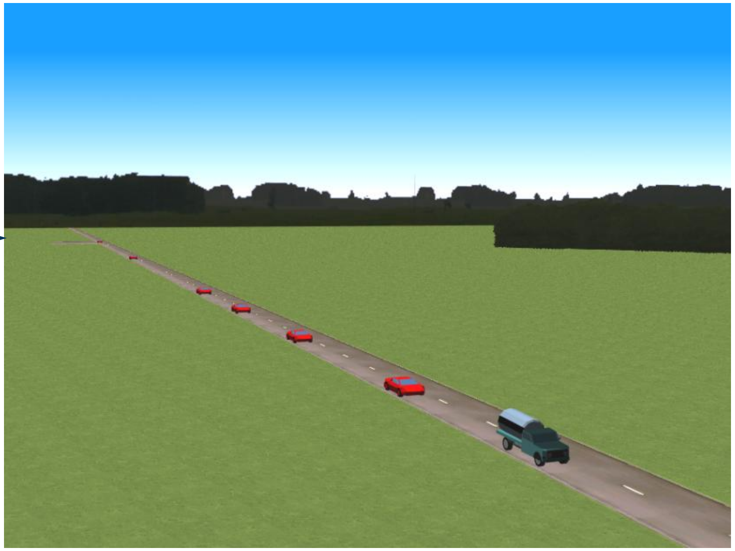
```
Execution begins

Objects created: 7 passive, 0 active   Pucks created: 1   Memory: 1,682,680   Time: 2.43 se
```

```
import <BaseEntity.slx>
import <slxrti10.slx>
module def
        {
 public type lsa_phase           enum(ROT,GELB,GRUEN);
type str_typ               enum(HS,GS,NS,SS,FS);
type stages                enum(G1,G2,G3,AN2,AN1,F1,F2,FF2,FF1);

constant boolean           priv_ON  = TRUE;
constant boolean           priv_OFF = FALSE;

constant float             EPS = 1E-4;

constant float             K1 = 8E-4;     // Konstante 1 fuer WSP
constant float             K2 = 4E-4;     // Konstante 2 fuer WSN
constant float             K3 = 6.5;      // Konstante 3 fuer WSP bzw. WSN
constant float             K4 = 0.3;      // Konstante 4 fuer WSP bzw. WSN

constant float             HALF_CHANAL = 2.0 ;
public boolean                   TRACE;          // TRUE = Animation

integer                    F_ID,P_ID;     // Counter fuer IDs
integer                    UPDATES;       // Anzahl der Ereignisse

constant float             RTI_delta = 0;   //Delta für Attribut-Updates //5

 public float                     SIM_TIME;      // Simulationszeit
public float                     STEADY_STATE;  // Steady-State-Zeitpunkt
public float                     TRACE_START;   // Startzeit fuer Schreiben des Trace-Files
```

```
Time: 0     Puck: (SYSTEM) 1/1  Status: moving  Priority: -2147483648     Line 18
```

SLX/Windows NT/PC

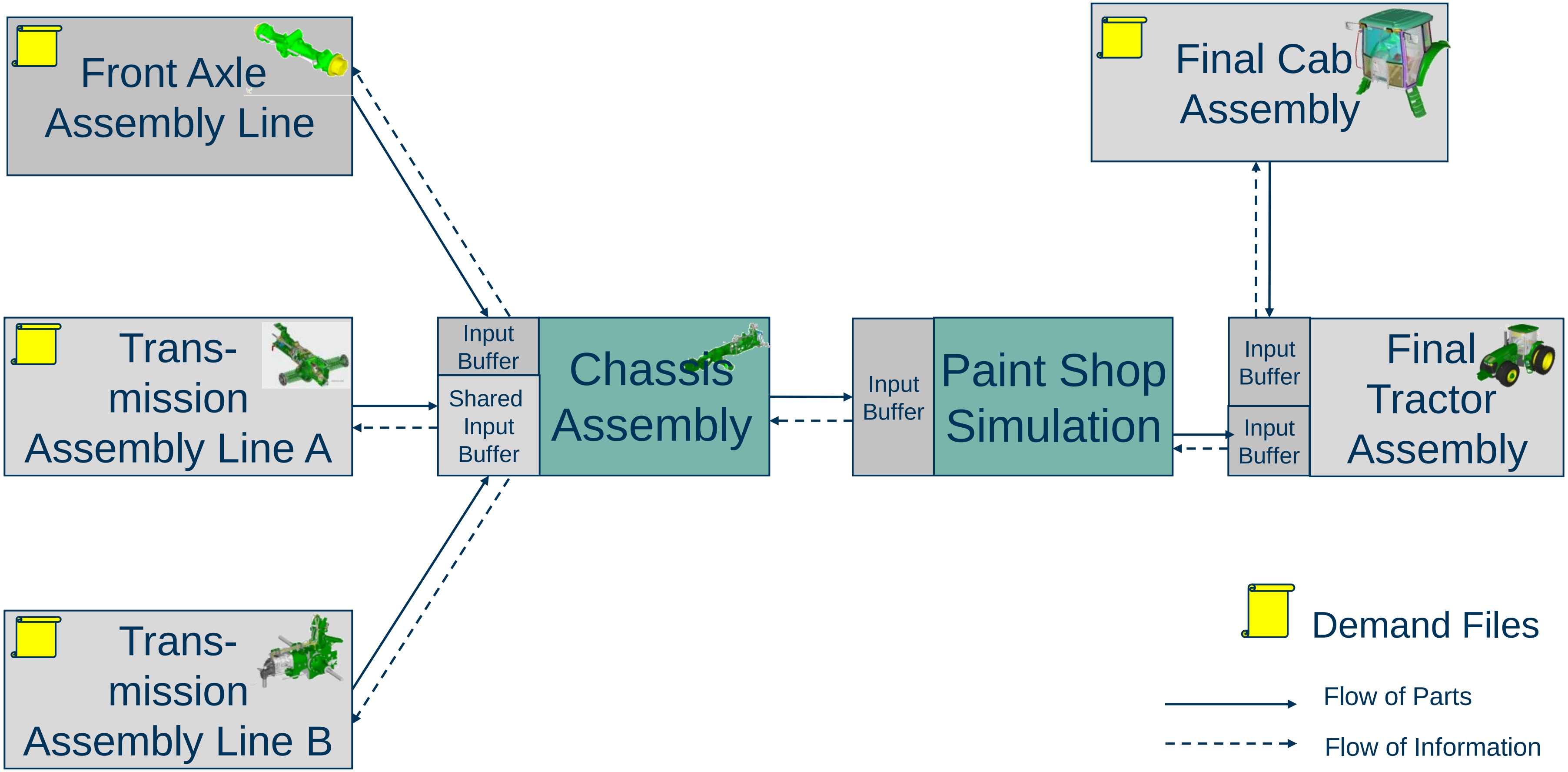
HLA



C++/SGI/Irix

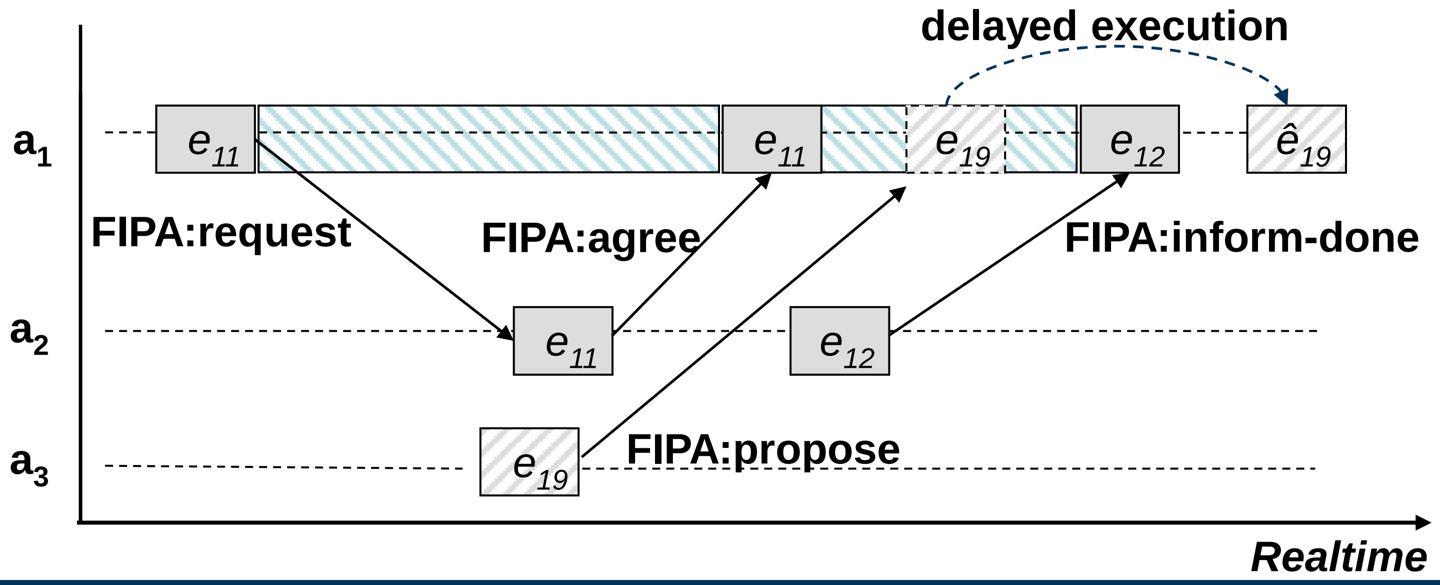
Case Study III – Digital Manufacturing

- Several scenarios combining DES-models



Case Study IV – Agent-based Simulation

- Test of synchronization algorithms for efficient distributed simulation of agent-based models
- Exploitation of properties of agent interaction patters to reduce risk of rollbacks in time warp



Contents

- State-of-the-Art in Distributed Simulation
- Experiences
  - Case Study “Supply Chain Simulation”
  - Case Study “Training Simulation”
  - Case Study “Digital Manufacturing”
  - Case Study “Agent-based Simulation”
- **Trends and Potential Impact**
- Summary

Lessons Learned

- Intelligent partitioning of models and efficient domain-specific synchronization algorithms are crucial for performance
  - For DES models, conservative synchronization is easily applicable (drawback: dependence on lookahead)
  - Optimistic synchronization promises better performance (if you can incorporate external knowledge)
  - Combined models need special consideration – do not try to distribute DE across models
- HLA is an industry strength DS standard – but there are potentially better solutions for massively parallel simulation requirements

Trends

- Efficient usage of GPU and multicore architectures
  - Huge potential, currently unused in COTS simulation packages
- Mobile devices
  - interface for life training exercises
  - New challenges: energy efficiency of algorithms, …

Potential Impact for Computational Systems Biology

- Multicore architectures (including GPU) offer huge potential – but require suitable algorithms and modeling paradigms
  - How to compartmentalize models at molecular level so that they can be efficiently simulated on a given architecture?
- Usability of existing synchronization algorithms?
  - For time stepped, round-based models there may be simpler, better scalable synchronization algorithms
  - Properties of modeling paradigm and interaction patterns should be taken into account
- Usability of existing technological solutions and standards?
  - HLA with its current RTI implementations is not too well suited for massively parallel models
  - Potential adaptability for massively parallel applications (“lightweight parallel HLA”)

Questions?

- Thank you for your attention!