

# Transformation-Based Component Concept & Integrated Modelling Techniques

Benjamin Braatz

TFS Research Colloquium  
June 6, 2002

---

## Generic Component Concept

Given a Class **Spec** of **Specifications** a **Component** *COMP* consists of

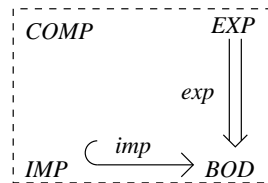
**Import Interface** *IMP*  $\in$  **Spec** with

**Import Connection** *imp*,

**Export Interface** *EXP*  $\in$  **Spec** with

**Export Connection** *exp* and

**Body** *BOD*  $\in$  **Spec**.



- **Constructor-Based Approach:** *imp* is a **Constructor**.
- **Transformation-Based Approach:** *exp* is a **Transformation**.

---

## Overview

- **Transformation-Based Component Concept** [FASE '02]
- **Formal Model of the Integration Paradigm** [GT-VMT '01]
- **Components of Integrated Specifications** [IDPT '02]
- **Current and Future Work** [GT-VMT '02], [Dipl. MK, MP, BB]

---

## Generic Transformation Concept

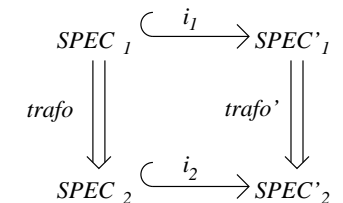
We assume a **Generic Framework**  $\mathcal{T}$  of **Transformations** *trafo*:  $SPEC_1 \Rightarrow SPEC_2$  satisfying the

**Extension Property:** For each Transformation *trafo*:  $SPEC_1 \Rightarrow SPEC_2$

with (consistent) Inclusion  $i_1: SPEC_1 \hookrightarrow SPEC'_1$

exists **Extension** *trafo'*:  $SPEC'_1 \Rightarrow SPEC'_2$

with Inclusion  $i_2: SPEC_2 \hookrightarrow SPEC'_2$ .



The **Transformation Semantics** of a Specification  $SPEC_1 \in \mathbf{Spec}$  is given by  $Trafo(SPEC_1) = \{ trafo: SPEC_1 \Rightarrow SPEC_2 \mid SPEC_2 \in \mathbf{Spec} \}$ .

---

## Framework of HLR-Transformations

The Generic Framework can be instantiated by **HLR-Systems**.

**Result [GT-VMT '02]:** HLR-Systems satisfy the Extension Property.

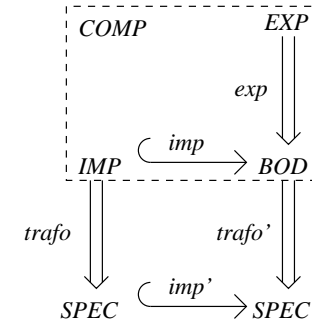
The Extension Property in this Case is a Consequence of the **Embedding Theorem**, where the Inclusion must be consistent with the Derivation. [PT '93]

We can assume that  $exp$  is a **Direct Derivation**, because the **Concurrency Theorem** ensures the Existence of a **Concurrent Production** for **Production Sequences** of Length  $n \geq 2$ . [EHKP '91]

## Transformation Semantics of Components

The **Transformation Semantics** of a Component  $COMP$  is given by

$$\begin{aligned} \text{TrafoSem}(COMP): \quad & \text{Trafo}(IMP) \rightarrow \text{Trafo}(EXP) \\ & (\text{trafo}: IMP \Rightarrow SPEC) \mapsto (\text{trafo}' \circ \text{exp}: EXP \Rightarrow SPEC'). \end{aligned}$$



## Composition of Components

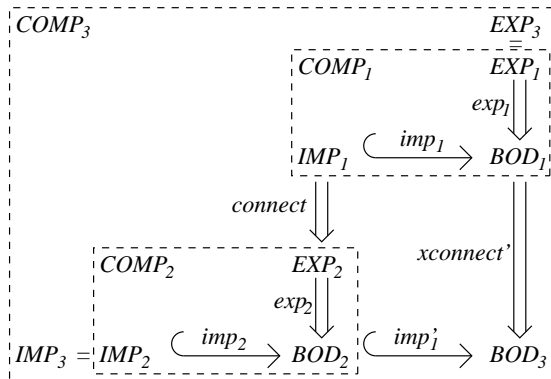
Given Components  $COMP_1$ ,  $COMP_2$  and **Connector**  $connect: IMP_1 \Rightarrow EXP_2$

the **Composition**

$$\begin{aligned} COMP_3 = \\ COMP_1 \circ_{connect} COMP_2 \end{aligned}$$

is defined by:

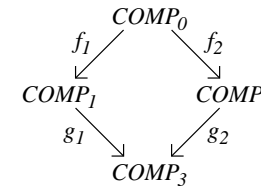
$$\begin{aligned} IMP_3 &= IMP_2 \\ EXP_3 &= EXP_1 \\ BOD_3 &\text{ by Extension} \\ imp_3 &= imp_1' \circ imp_2 \\ exp_3 &= xconnect' \circ exp_1 \end{aligned}$$



**Result [FASE '02]:** Transformation Semantics is compositional.

## Union of Components

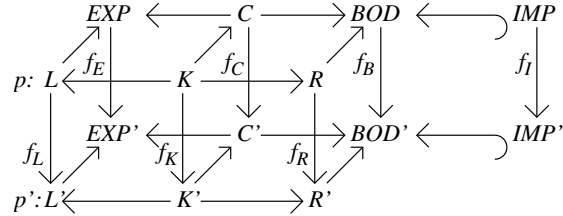
**Union** of Components  $COMP_1$  and  $COMP_2$  via a shared Sub-Component  $COMP_0$  is a Pushout in a suitable **Category of Components**.



**Component Morphisms** must be compatible with the Export and Import Connections.

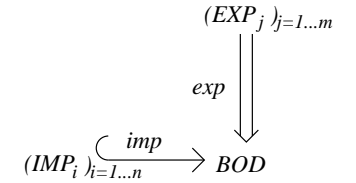
## Morphisms for HLR-Transformation Components

A **HLR-Component Morphism**  $f: COMP \rightarrow COMP'$  consists not only of Morphisms  $f_E$ ,  $f_B$  and  $f_I$ , translating Export, Body and Import, but also of Morphisms  $f_L$ ,  $f_K$  and  $f_R$ , translating the **Production**  $p$  of  $exp$ , and a Morphism  $f_C$ , translating the **Context Object**  $C$  of  $exp$ .



## Multiple Interfaces

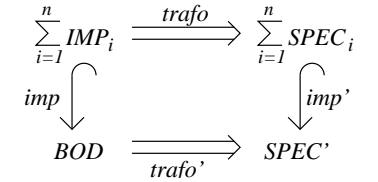
**Multi-Interface Components** are allowed to have **Families of Import and Export Interfaces** and corresponding **Families of Connections**.



To construct the Transformation Semantics we need the **Existence** of

a **unique Inclusion**  $imp$  and a **unique Transformation**  $trafo$

in the following Extension Property Diagram.



Multi-Interface Components allow **Partial Composition**.  
 $\implies$  Architectural Approach of Fiadeiro Group

## Integrated Signatures and Models

The **Model Functor**  $Mod: \text{IntSig} \rightarrow \text{Cat}$  assigns a Category of **Integrated Models** to each **Integrated Signature** in the Category **IntSig**.

Layer	Integrated Signatures	Integrated Models
<b>Data Type</b>	$\Sigma_0 \in \mathbf{Sig}$	$A_0 \in PAlg(\Sigma_0)$
<b>Data States</b>	$\Sigma \xleftrightarrow{in} \Sigma_0 \in \mathbf{Sig}$	$DS \subseteq \{A \in PAlg(\Sigma) \mid V_{in}(A) = A_0\}$
<b>Transformations</b>	$T \in \mathbf{Set}$ $dom: T \rightarrow S^*$ $codom: T \rightarrow S^*$	$TS \subseteq \{t_{A,B}(a,b) \mid t \in T, A, B \in DS, a \in A_{dom(t)}, b \in B_{codom(t)}\}$ $G_{DSTS} = (DS, TS, src, trg) \in \mathbf{Graph}$ with $src(t_{A,B}(a,b)) = A$ and $trg(t_{A,B}(a,b)) = B$
<b>Processes</b>	$P \in \mathbf{Set}$ $dom: P \rightarrow S^*$ $codom: P \rightarrow S^*$	$PR = (G_{RSTS}(p), h(p))_{p \in P}$ $G_{RSTS}(p) \in \mathbf{Graph}$ $h(p) = (h(p)(a,b): G_{RSTS}(p) \rightarrow G_{DSTS})_{a \in A_{dom(p)}, b \in B_{codom(p)}}$

## Instantiations of the Integration Paradigm

A **Modelling Technique** consisting of a Class of High-Level Structures  $\mathcal{HL}$  is **interpreted** in the Integration Paradigm by  $\mathcal{HL}\text{-Sig}: \mathcal{HL} \rightarrow \text{IntSig}$  and  $\mathcal{HL}\text{-Mod}: \mathcal{HL} \rightarrow \text{Cat}$  with  $\mathcal{HL}\text{-Mod}(HL) \subseteq Mod(\mathcal{HL}\text{-Sig}(HL))$ .

**Data Type Modelling Techniques:** (Conditional) Equational Logic, First Order Logic

**Process Modelling Techniques:** Place/Transition Petri Nets [TR 2001/21], Graph Transformation [TR 2001/21], CCS and other Process Algebras (Sketch), Statecharts (Open)

**Integrated Modelling Techniques:** Algebraic High-Level Petri Nets [TR 2001/21], Attributed Graph Transformation [TR 2001/21], Unified Modeling Language (Open), LOTOS (Open)

---

## Modelling Techniques as High-Level Constraints

A Modelling Technique  $\mathcal{HL}$  with an Interpretation  $(\mathcal{HL}\text{-Sig}, \mathcal{HL}\text{-Mod})$  constitutes a **Logic of  $\mathcal{HL}$ -Constraints**  $(\mathcal{HL}\text{-Constr}, \models_{\mathcal{HL}})$  with  $\mathcal{HL}\text{-Constr}(I\Sigma) = \{(HL, h) \mid HL \in \mathcal{HL}, h: \mathcal{HL}\text{-Sig}(HL) \rightarrow I\Sigma\}$ ,  $\mathcal{HL}\text{-Constr}(f)(HL, h) = (HL, f \circ h)$  for  $f: I\Sigma \rightarrow I\Sigma'$ ,  $(HL, h) \in \mathcal{HL}\text{-Constr}(I\Sigma)$  and  $M \models_{\mathcal{HL}} (HL, h) \iff \text{Restr}_h(M) \in \mathcal{HL}\text{-Mod}(HL)$ .

### The Satisfaction Condition

$M' \models_{\mathcal{HL}} \mathcal{HL}\text{-Constr}(f)(HL, h) \iff \text{Restr}_f(M') \models_{\mathcal{HL}} (HL, h)$  can easily be shown to hold for all  $f: I\Sigma \rightarrow I\Sigma'$ ,  $(HL, h) \in \mathcal{HL}\text{-Constr}(I\Sigma)$  and  $M' \in \text{Mod}(I\Sigma')$ .

---

---

## Transformation-Based Integrated Components

An **Integrated Specification**  $ISP = (ISP_{Sig}, ISP_{Cons})$  consists of an Integrated Signature  $ISP_{Sig} \in \text{IntSig}$  and a Constraint  $ISP_{Cons} \in \mathcal{HL}\text{-Constr}(I\Sigma)$ .

### Version 1:

$imp$  is an **Inclusion of the Signature and the Constraint** of  $IMP$ .

$exp$  is a **Transformation of the Signature and the Constraint** of  $EXP$ .

**Internal Correctness** is given by

$IMP_{Cons} \subseteq BOD_{Cons}$  and  $exp(EXP_{Cons}) = BOD_{Cons}$ .

$\implies$  Transformation Framework for Constraints needed.

### Version 2:

$imp$  is only an **Inclusion of the Signature** of  $IMP$ .

$exp$  is only a **Transformation of the Signature** of  $EXP$ .

**Internal Correctness** is given by  $(imp(IMP_{Cons}) + BOD_{Cons}) \longrightarrow exp(EXP_{Cons})$ .

$\implies$  Deduction Mechanism for Constraints needed.

---