



Workshop on Cooperative Information Systems :  
The Role of Agents and Goals  
Toronto, Canada, September 19-21, 1999

## Agent-oriented RE with Albert II

M. Petit & the Albert research group  
University of Namur  
Belgium

[Http://www.info.fundp.ac.be/albert](http://www.info.fundp.ac.be/albert)



## Outline

- Part I : Background
  - The Albert II RE language
- Part II : Recent and On-going Research
  - i\* / Albert combination
  - A multiformalism & component-based approach to the RE of CIM systems
  - Cooperative animation and integration in a scenario- and goal-based approach to RE
  - Reasoning assistant
- Part III : Future Work



## Part I : Background The Albert II RE language



## Milestones

- Albert in 1992
  - Result of the ICARUS Esprit II project
  - Internal validation on academic case studies
  - Intuitive semantics
- Albert II in 1995
  - Internal validation on real-size case studies
  - Semantics based on a variant of RT-OSL (Böhm & Sernadas, 1993)
- Albert II (version 2) in 1997
  - External validation on real projects (TELECOM,CIM)
  - New semantics
  - Development of tools

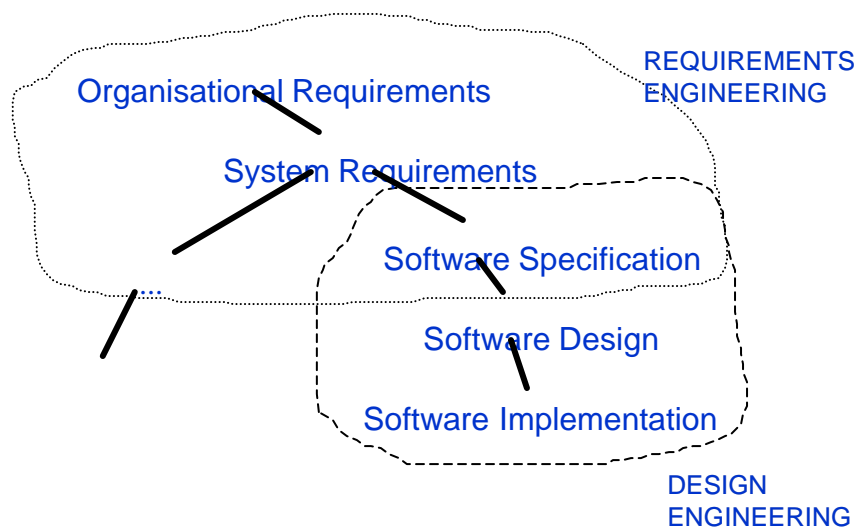


## The Projects

- In the past...
  - ICARUS
  - 2RARE
  - Pégard, ... (CIM)
  - MODELAGE
- Currently...
  - CAT
  - CREWS
  - RENOIR
  - ASPIRE
  - FIREWORKS
  - Cooperative IS
  - AgentLink



## Focus on Late RE





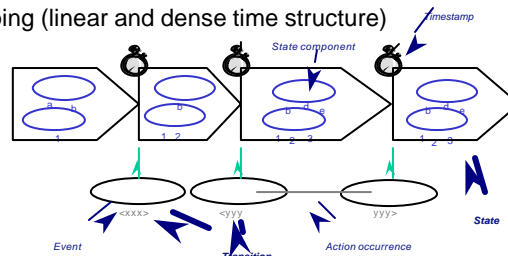
## Properties of Requirements Modelling Languages

- Precise (formal, clear semantics)
- Expressive
- Natural
- Structured
  - ease of modifications
- Adequate to the application domain
- Implementation-independent



## Albert II, a Requirements Modelling Language for Real-Time and Distributed Systems

- An **agent-oriented** (~ object-oriented) paradigm
  - A (**composite**) **system** is composed of **cooperating agents** which are responsible for **actions** and **state components**
- Formal semantics based on a **real-time temporal logic** with strong typing (linear and dense time structure)



- Designed with concerns of **expressiveness** (richness of the underlying models) and **naturalness** (straightforward mapping of customers' statements)

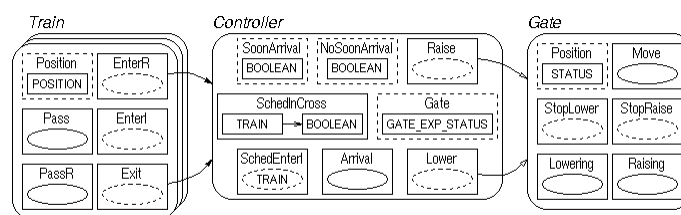


## Structure of an Albert II spec

- An Albert II spec is made of :
  - a description of the **system's structure** (**individual agents**, **classes of agents** or **agent societies**)
  - definition of agents (or agent classes)
    - **state components** and **actions** declarations
    - **constraints**
  - definition of **data type** and **opérations**



## Snapshot of an Albert II Specification



### BASIC CONSTRAINTS

INITIAL VALUATION

Gate = Up

SchedInCross[t] = FALSE

SoonArrival = FALSE

NoSoonArrival = TRUE

### LOCAL CONSTRAINTS

STATE BEHAVIOUR

SoonArrival  $\iff$

$\exists t: (\neg \text{SchedInCross}[t])$

$\wedge \Diamond_{\geq \text{GammaDown}} \text{SchedInCross}[t]$

NoSoonArrival  $\iff$

$\forall t: (\neg \text{SchedInCross}[t])$

$\wedge \square_{\geq \text{GammaUp} + \text{Delta} + \text{GammaDown}} \neg \text{SchedInCross}[t]$

### EFFECTS OF ACTIONS

SchedEnterI(t): SchedInCross[t] := TRUE

t.Exit: SchedInCross[t] := FALSE

Lower: Gate := Down

Raise: Gate := Up

### CAPABILITY

$\mathcal{X}\mathcal{O}(\text{Raise} / \text{Gate} = \text{Down} \wedge \text{NoSoonArrival})$

$\mathcal{X}\mathcal{O}(\text{Lower} / \text{Gate} = \text{Up} \wedge \text{SoonArrival})$

$\mathcal{F}(\text{Arrival}(t) / \text{SchedInCross}(t))$

### ACTION COMPOSITION

Arrival(t)  $\iff$  t.EnterR ; SchedEnterI(t)

### ACTION DURATION

| Arrival | = Epsilon1

### COOPERATION CONSTRAINTS

ACTION PERCEPTION

$\mathcal{K}(t.\text{EnterR} / \text{TRUE})$

$\mathcal{K}(t.\text{Exit} / \text{TRUE})$

ACTION INFORMATION

$\mathcal{K}(\text{Raise}.g / \text{TRUE})$

$\mathcal{K}(\text{Lower}.g / \text{TRUE})$



## Expressiveness and Naturalness

- Guidance of the requirements engineer in his task by providing **templates for typical properties**
- Supports a mixed style of specification (**declarative** and **operational** styles)
  - **Declarative constraints**  
«If a train is at the sensor then it will eventually be in the crossing» \_\_\_\_\_
  - **Operational constraints**  
«The gate must be lowered 15" after a train has passed the entrance sensor»
- Reasoning about **communication**
  - «The gate does not perceive an order to lower emitted by the controller if it is not up»
- Supports the expression of typical **distributed** and **real-time systems'** properties
  - **Modelling real-time performances**  
«Lowering the gate takes between 0 (strictly) and 20 seconds»



## Benefits and Challenges

- Benefits
  - formality enforces precision
  - structuring & constraint templates are a first step towards application of a formal language for RE
- Challenges
  - progressive specification elaboration and evolution
    - *link with earlier RE phases (product & processes)*
    - *domain-specific methodology*
  - help the stakeholders gain confidence in specs
    - *need validation & verification tools*



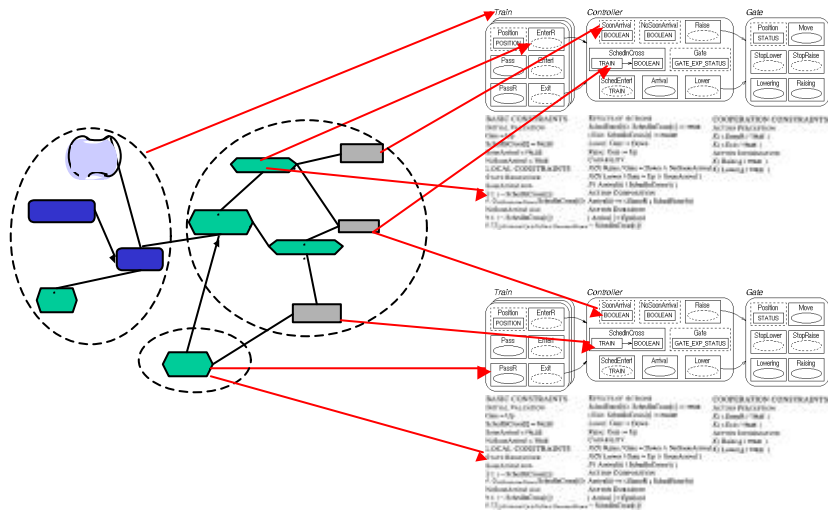
## Part II : Recent and On-going Research



### II.1 Combination of $i^*$ & Albert II



## Combination of $i^*$ & Albert II



## Albert/ $i^*$ Combination Benefits

- Progressive Albert specification elaboration
- Show operationalization of  $i^*$  elements
- Traceability for Albert specification elements
- Link to NFR
- Representation of alternatives (link several specs)



## II.2 A multi-formalism & component-based approach to the RE of CIM systems



### CIM Multi-formalism Approach

	Expressiveness									
	Function	Information	Resource	Organization	NFR	Rationales	Naturalness	Precision		Reuse
CIMOSA	😊	😊	😊	😊	😊	😊	😊	😊	😊	Standardized domain-specific concept for CIM
<i>i*</i>	😊	😊	😊	😊	😊	😊	😊	😊	😊	Goals, rationales, NFR
Albert II	😊	😊	😊	😊	😊	😊	😊	😊	😊	Precision, expressiveness
Telos	😊	😊	😊	😊	😊	😊	😊	😊	😊	Integration, reuse

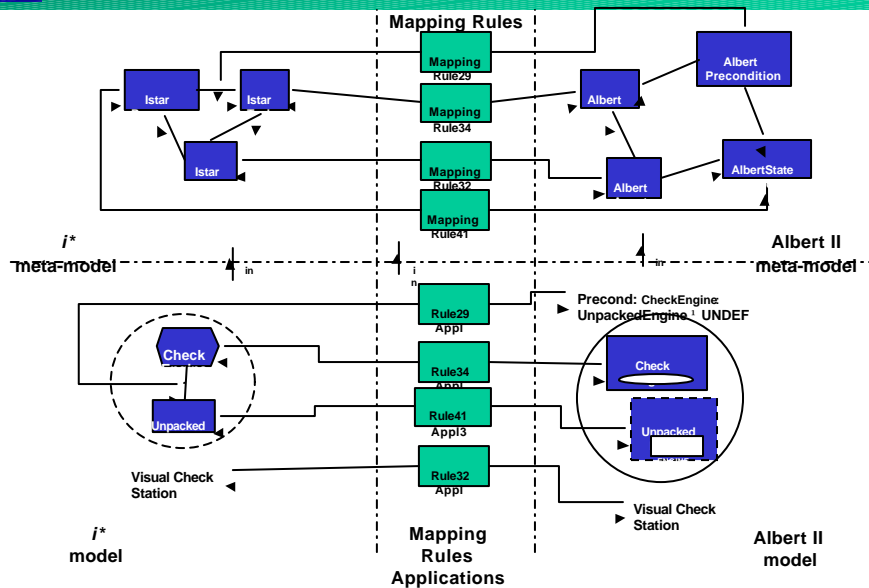


# CIM Multi-formalism Approach

- Combination of languages
  - loose integration versus strong integration
  - all languages are used together where most appropriate
  - they can be used independently
- When needed, a corresponding model expressed in a different language can be created
  - to gain precision
  - to gain expressiveness
  - to gain naturalness
- Depending on modeling needs
  - not complete (only parts)
  - not automatic (user decide what and how to map)



# Mapping Rules



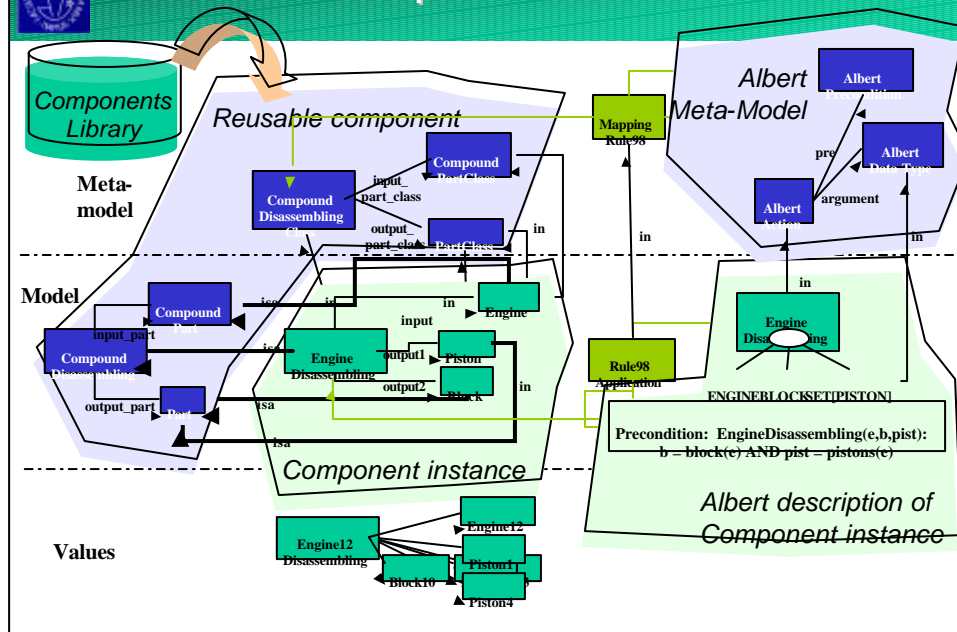


# Mapping Rules

- Mapping rules
  - defined at the language level (meta-model)
  - relate concepts of two languages
  - defined once for a pair of languages
- Applications of mapping rules
  - at the model level
  - relate elements of two models in two different languages
  - A MR can be applied several times on different elements of a single model
- Telos is used for
  - expressing the meta-models
  - describing the mapping rules
  - recording their applications
- Recording the application of rules allows:
  - navigating among models written in different languages
  - traceability



# Components for CIM



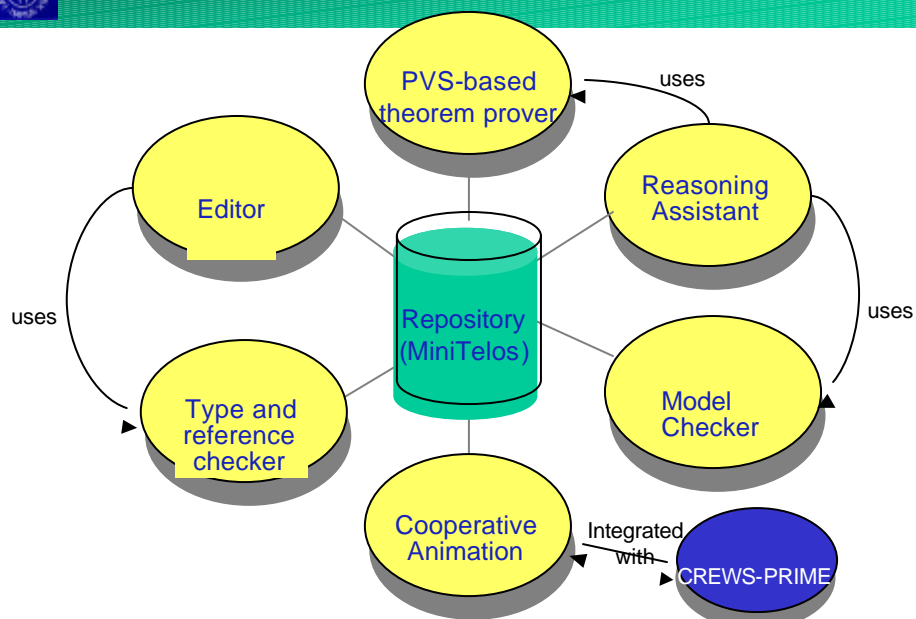


## CIM Components Library

- Static aspects
  - Part (basic, compound parts, features, position, identity, existing parts, buffers)
- Behavioral aspects
  - Functions (based on characteristics of parts)
  - Control (when functions are triggered)
- Functional goals
  - manufacturing order
- Entities (Agents)
  - physical manufacturing role, controller role, customer role, concrete manufacturing entity



## Tools : Overview





## II.3 Cooperative animation and integration in a scenario- and goal-based approach to RE



### The Albert II - CREWS Animator

#### Goals

**G1 - Validation problem:**

*How to make sure the right system is built?  
How to detect errors early?*

**G2 - Understanding problem:**

*How to envision scenarios allowed and disallowed by a requirements specification?*

**G3 - Involvement problem:**

*How to make people participate to requirements validation?*

**G4 - Cost problem:**

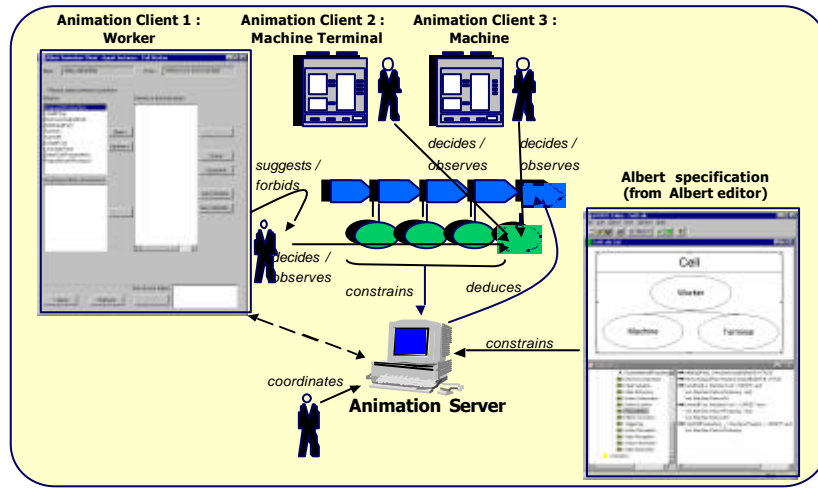
*How to reduce costly prototype development or heavy formal analysis techniques?*



# The Albert II - CREWS Animator

## Overview of approach :

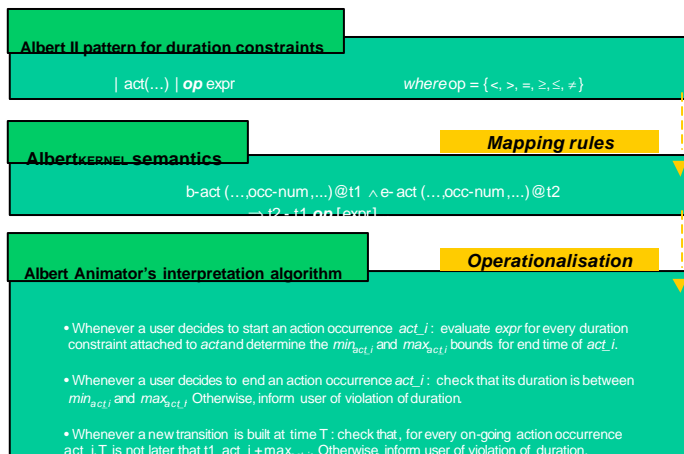
- Discovering problems by *testing specifications against scenarios* (G1 & G2)
- *Cooperative* and *interactive* (G3)

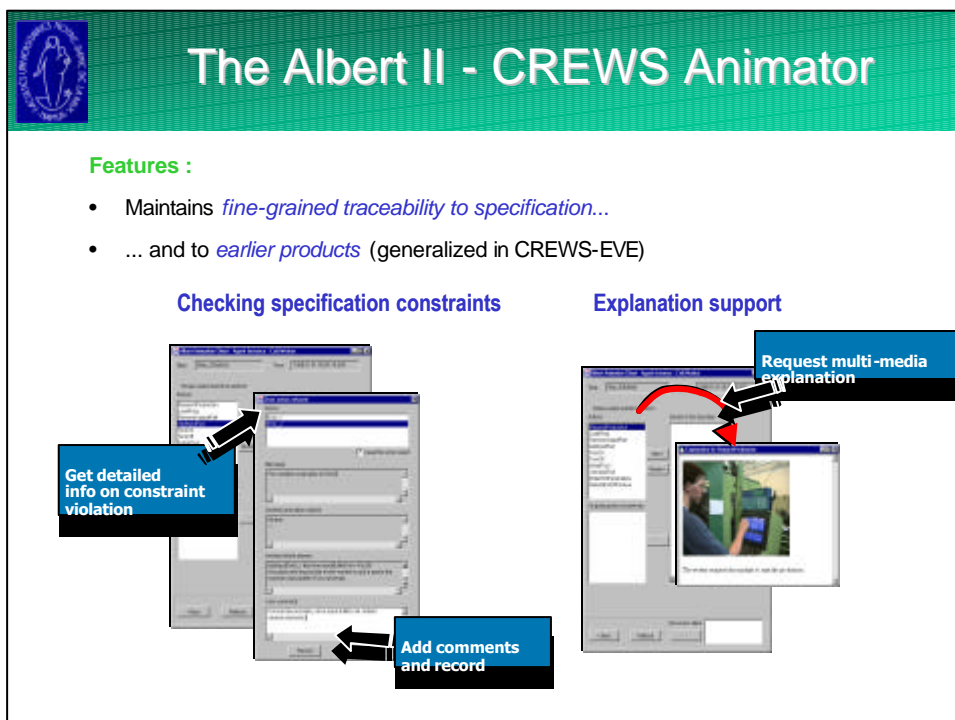
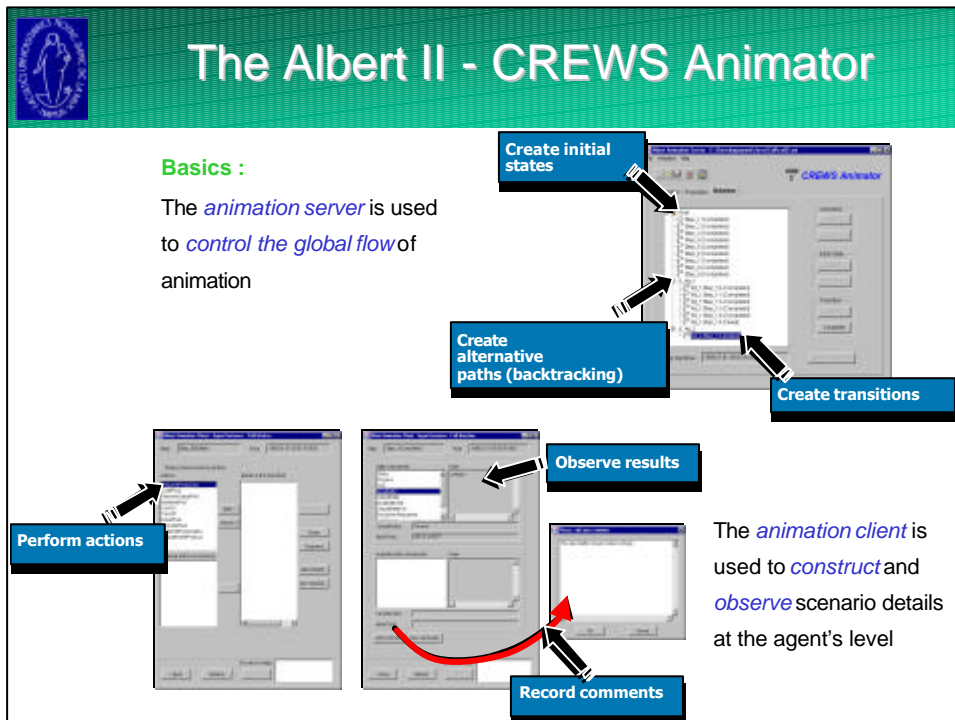


# The Albert II - CREWS Animator

## Overview of approach (cont's):

- Interpretation with partial hard-coding : *transparent & efficient* (G4)



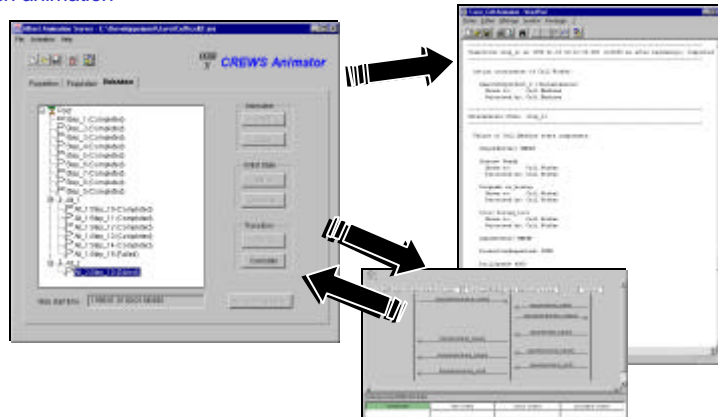




## The Albert II - CREWS Animator

### Features (cont's):

- *Trace generation* :
  - MSCs as graphical traces (through CREWS-EVE)
  - detailed textual traces (incl. problems discovered and comments)
- *Batch animation*



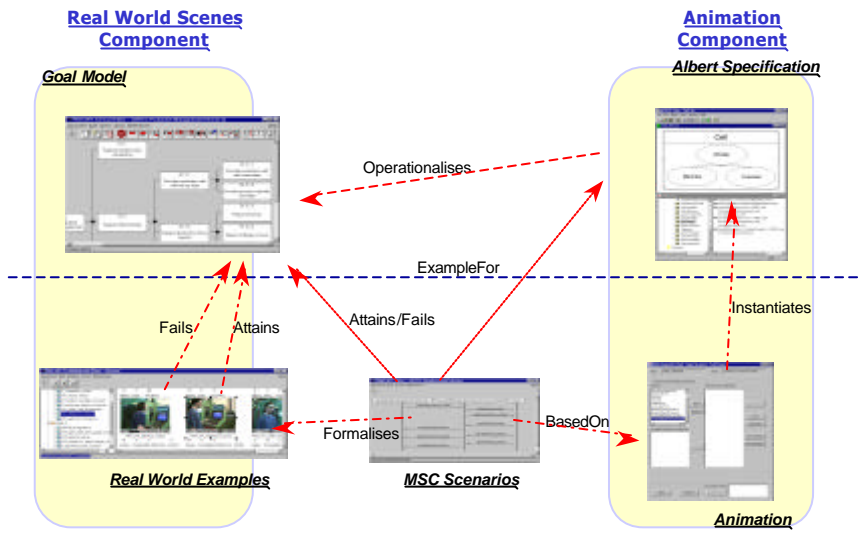
## The Albert II - CREWS Animator

### Benefits :

- The specification is understood through the *scenarios*
- Every stakeholder participates to validation in a *focused, interactive* way
- Co-operative animation *triggers communication* between stakeholders
- *Unexpected situations are discovered* and scenarios can be reused as *test cases*
- *No need to build a prototype*

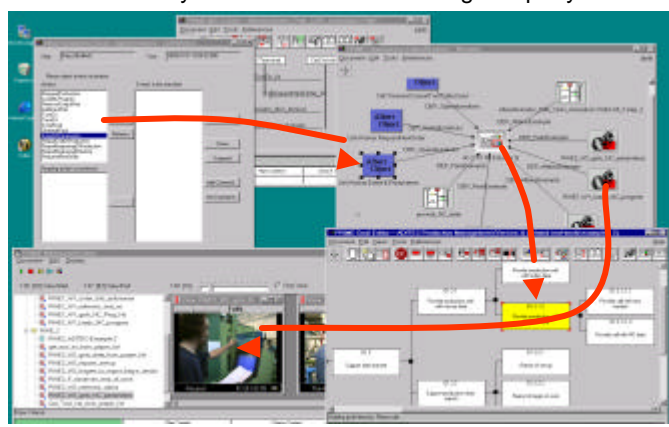
# CREWS-EVE: Process-Integrated Tool Components Providing Full Traceability

In collaboration with RWTH-Aachen



# CREWS-EVE method chunks

- Guided method chunks implemented so far:
  - Validation and elicitation of goals (and their operationalisations) using animation
  - Validating Albert II specifications by reconstructing real world scenarios
  - Explanation support for animation steps
- Source : Case-study at ADITEC manufacturing company

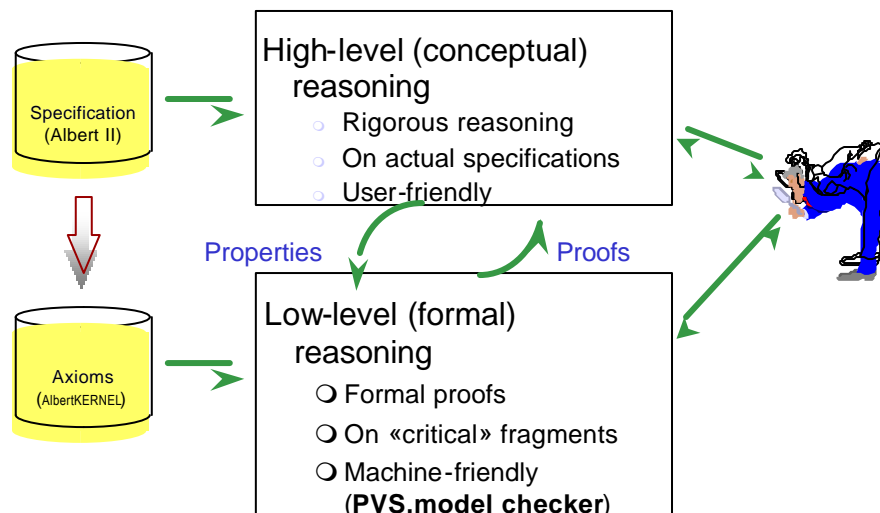




## II.4 Reasoning Assistant

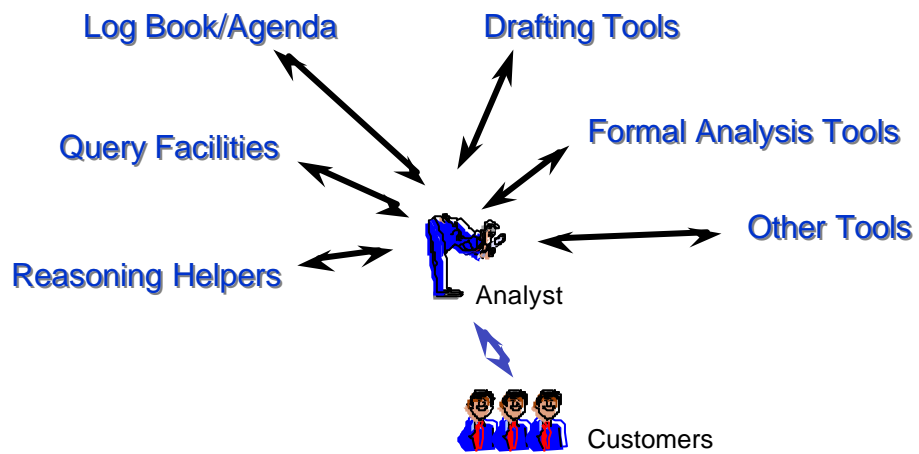


## 'High and Low-Level' Reasoning





## Tools of a Conceptual Reasoning Assistant



## Part III : Future Research Directions



## Future Research Directions

- Pursue tool development (especially high- and low-level reasoning)
- Tool evaluation and dissemination
- Implement mapping rules between goals and Albert as method chunks in CREWS-EVE
- Probabilistic extensions to Albert II for risk assessment (for medical systems e.g. PMS)
- Problem frames and formal methods



## Problem frames [Jackson95] and Formal Methods

- “Problem frame” = fundamental structure of the problem at hand
- Efficient method = method specific to a problem frame
  - Recurring problem frames : IS frame, control frame, connection frame, workpieces frame, ...
  - Actual problems are multi-frames problems
- Our objective : provide support for problem frames frequent in critical systems :
  - specialised languages (on top of Albert)
  - reusable methodological support
  - tools