Enterprise Modelling: Objectives, constructs & ontologies

F.B. Vernadat

LGIPM, University of Metz, France

Eurostat, European Commission

Tutorial held at the EMOI-CAiSE Workshop

Riga, Latvia, June 7, 2004

Short Bibligraphy

ESPRIT Consortium AMICE, CIMOSA: Open System Architecture for CIM, 2nd edition, Springer-Verlag, 1993.

Petrie, C. (ed.) Enterprise Modeling Technology, The MIT Press, 1992.

Vernadat, F.B. Enterprise Modeling and Integration: Principles and Applications, Chapman & Hall, 1996.

Bernus, P., Mertins, K., Schmidt, G. (eds.) Handbook of Information Architectures, Springer-Verlag, 1998.

Kosanke, K. and Nell, J. (eds.) Enterprise Engineering and Integration: Building International Consensus, Springer-Verlag, 1997.

Molina, A., Kusiak, A, Sanchez, J. (eds.) Handbook of Life Cycle Engineering, Kluwer Academic Pub., 1998.

Kosanke, K. et al. (eds.) Enterprise Inter- and Intra-Organizational Integration: Building International Consensus, Kluwer Academic Pub., 2003.

Contents

- Enterprise Modelling
  - Why?
  - What?
  - How?
- Modelling Constructs
  - Function View
  - Information View
  - Resource View
  - Organisation View
- Enterprise Ontologies
  - CIMOSA Ontology
  - TOVE Ontology / PSL
  - Enterprise Ontology

Part I

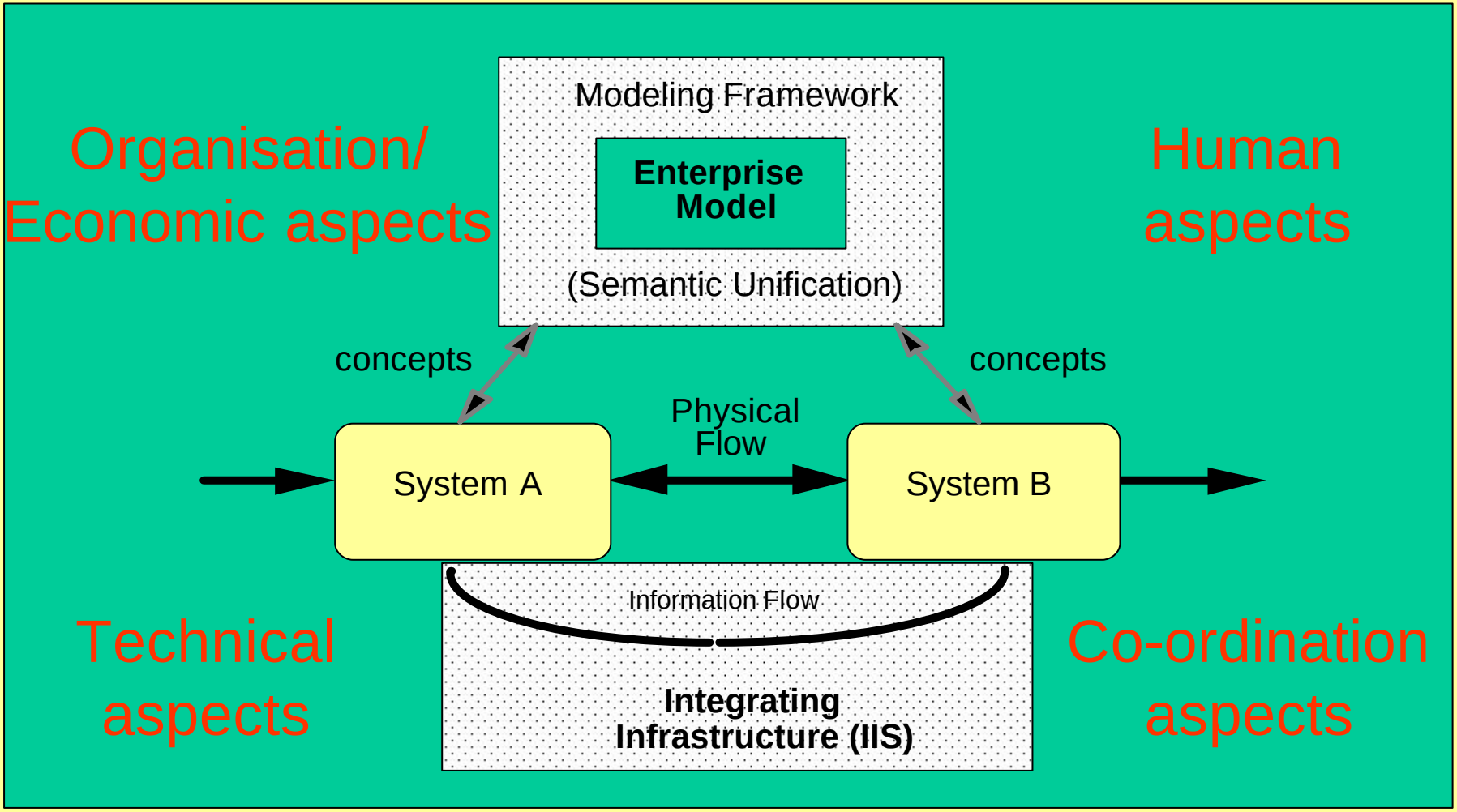
Enterprise Modelling Objectives

Enterprise Modelling: Why?

Rationale: The highly competitive global economy forces companies to:

- Fully understand and harness the way they operate
- Align their organisation structure with changing business needs
- Integrate enterprise networks (Extended Enterprises, Virtual/Agile Enterprises, Supply Chains, …)
- Implement large interoperable information systems (e.g. MES, ERP, PDM, SCM, …)
- Continuously optimise their operations, facilities and management in terms of Quality, Costs & Delays (QCD)

Enterprise Modelling: Why?



Enterprise Modelling: Why?

Major drivers for EM applications:

- Diagnosis of a disorder (material, information or control flows)
- Restructuring a business entity (to improve its performances)
- Business Process Reengineering (BPR)
- Large scale systems integration / interoperability
- Implementation of MES, ERP, PDM or APS systems
- Tuning the organisation structure to face business change
- Alignment or conformity to norms (ISO 9000, ISO 14000)
- Management decision (activity externalisation/internalisation)

Enterprise Modelling: What?

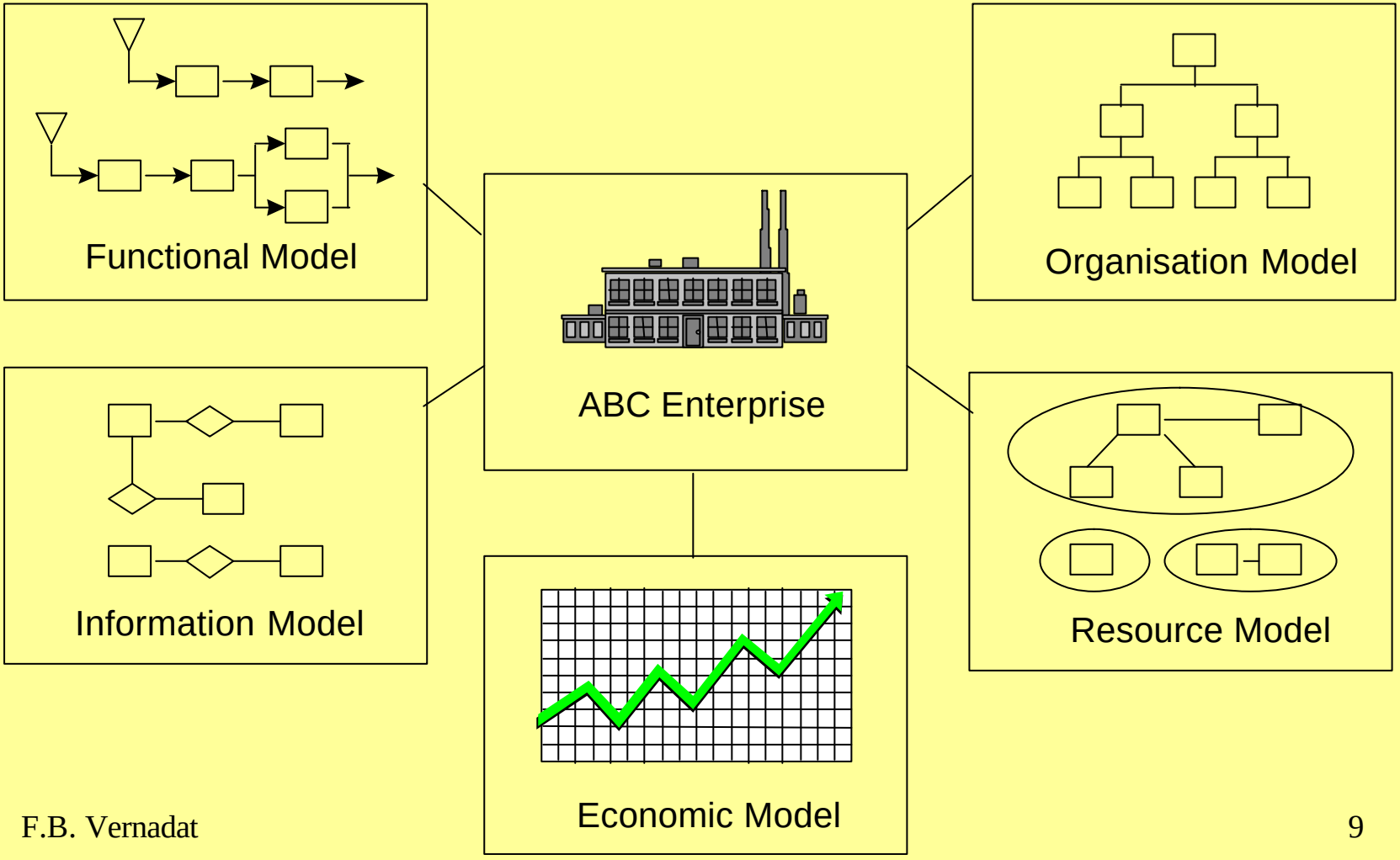
Definition: Enterprise Modelling (EM):

- The art of externalising knowledge which adds value to the enterprise or needs to be shared, i.e. to describe the things of the enterprise
- Concerns function, behaviour, information, resource, organisation, economic or other aspects of a business entity
- Used to represent the structure, behaviour, components and operations of a business entity to understand, (re)engineer, evaluate, optimise and even control business operations and performance

Must be open to simulation/analysis and decision support

Enterprise Modelling: What?

Enterprise model: not one monolithic model but an assemblage of models



Enterprise Modelling: What?

Added value of EM:

The four essential goals of Modelling, i.e.:

- to understand/explain
- to experiment (analyse, compare, test, evaluate/predict performances)
- to learn & decide (what-if scenarios)
- to operate/control

govern developments in Enterprise Modelling

Major advantage: to build a common consensus on how enterprise operations work (or should work)

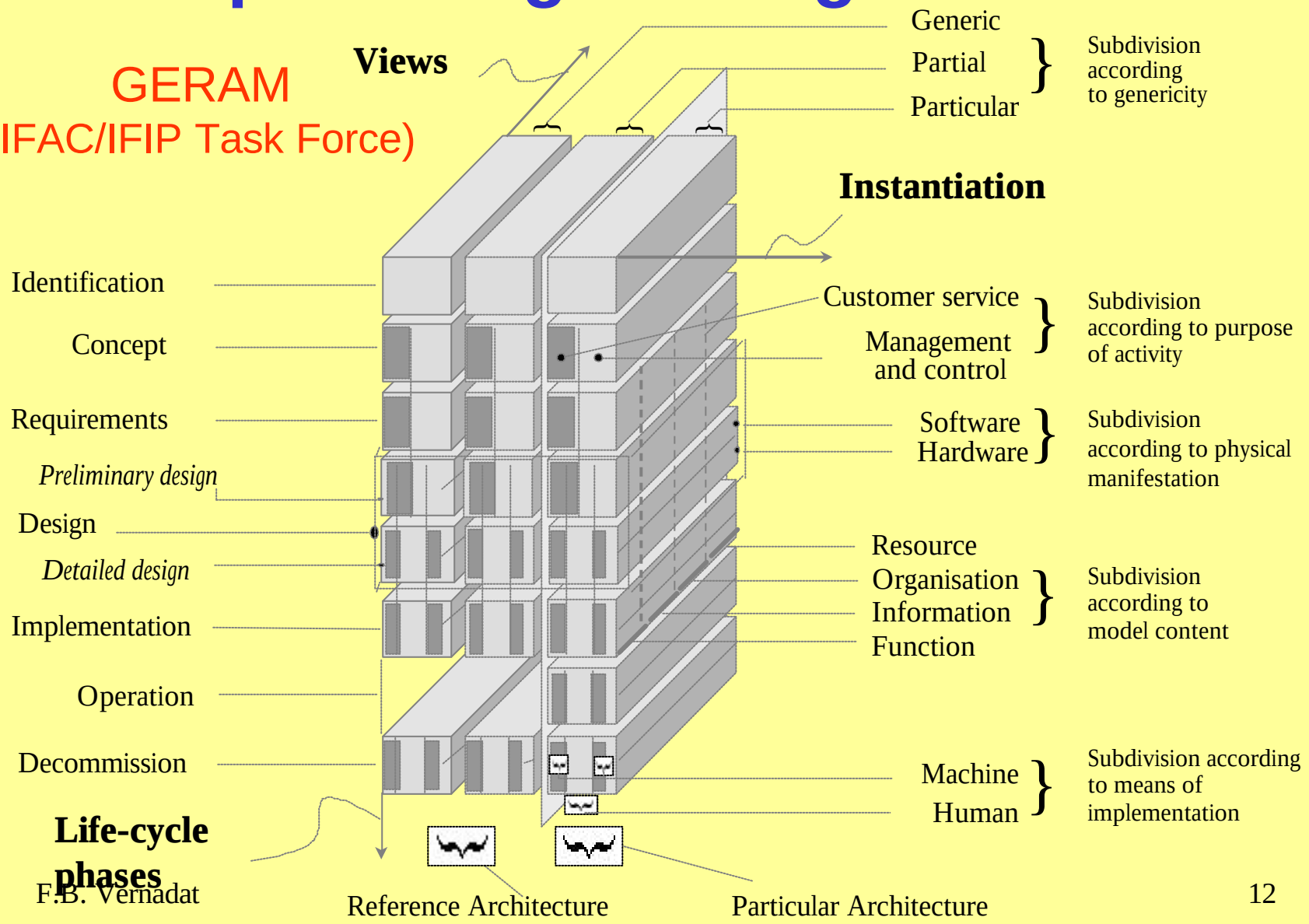
Enterprise Modelling: What?

Brief EM history & background:

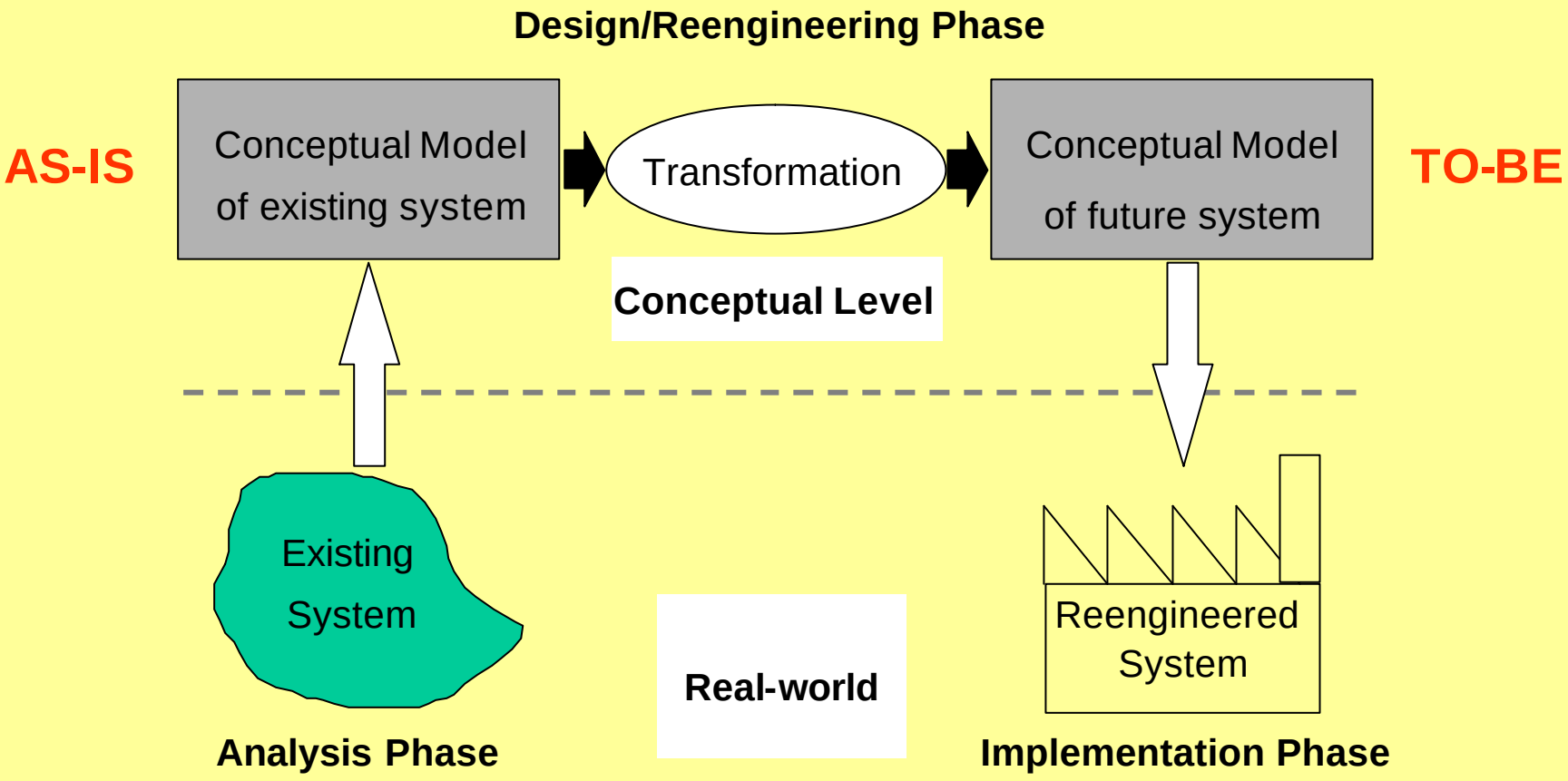
- Mid-70's: SADT, ER model, DFD, semantic nets (IT appli. dev't)
- 80's: CIM methods (ICAM-IDEF, GRAI, CAM-I)
- Late 80's: EU AMICE's CIMOSA / BPR (process orientation)
- Over the 90's:
  - ERP deployment (DEM, ARIS, …)
  - Workflow management systems (WPDL)
  - Object orientation (IEM, UML, …)
  - Ontologies (IDEF5, TOVE, Enterprise Ontology, PSL, i*)

Enterprise Engineering: Ref. Archi.

GERAM
(IFAC/IFIP Task Force)



Enterprise Modelling: How?



Rule: Must rely on a participative approach

Enterprise Modelling: How?

EM Good News:

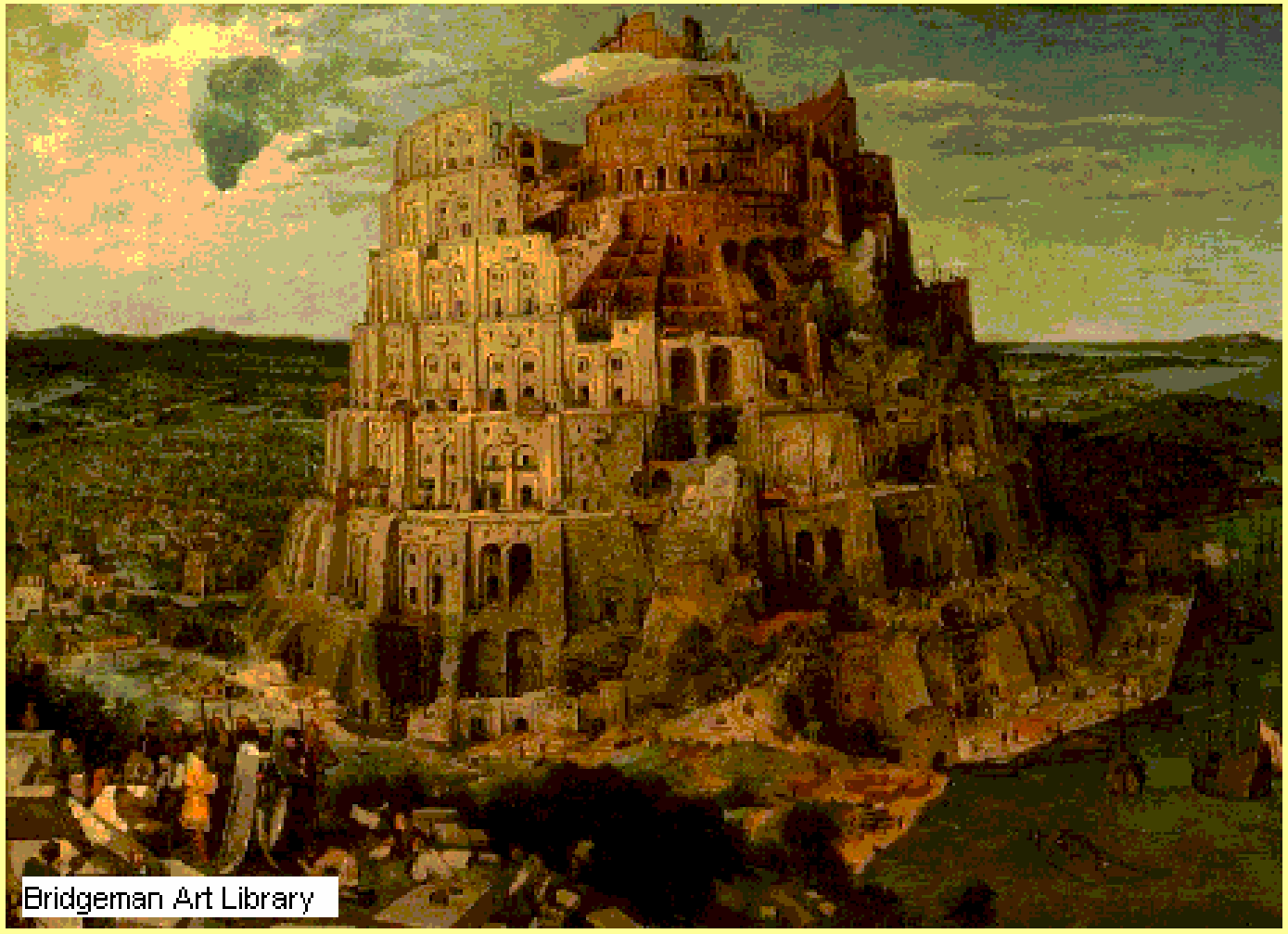
- Many commercial EM tools:

  ARIS Toolset, FirstSTEP, Bonapart, KBSI tools, PrimeObject, Enterprise Modeler, MO2GO, emaGIM, CimTool, …

- Many Workflow Management tools:

  IBM Flow Mark, Oracle Workflow, Ultimus, WorkParty, Ensemble, InConcert, Action Workflow, OPEN/Workflow, Staffware, Lotus Notes, …

EM bad news:
Tower of Babel situation



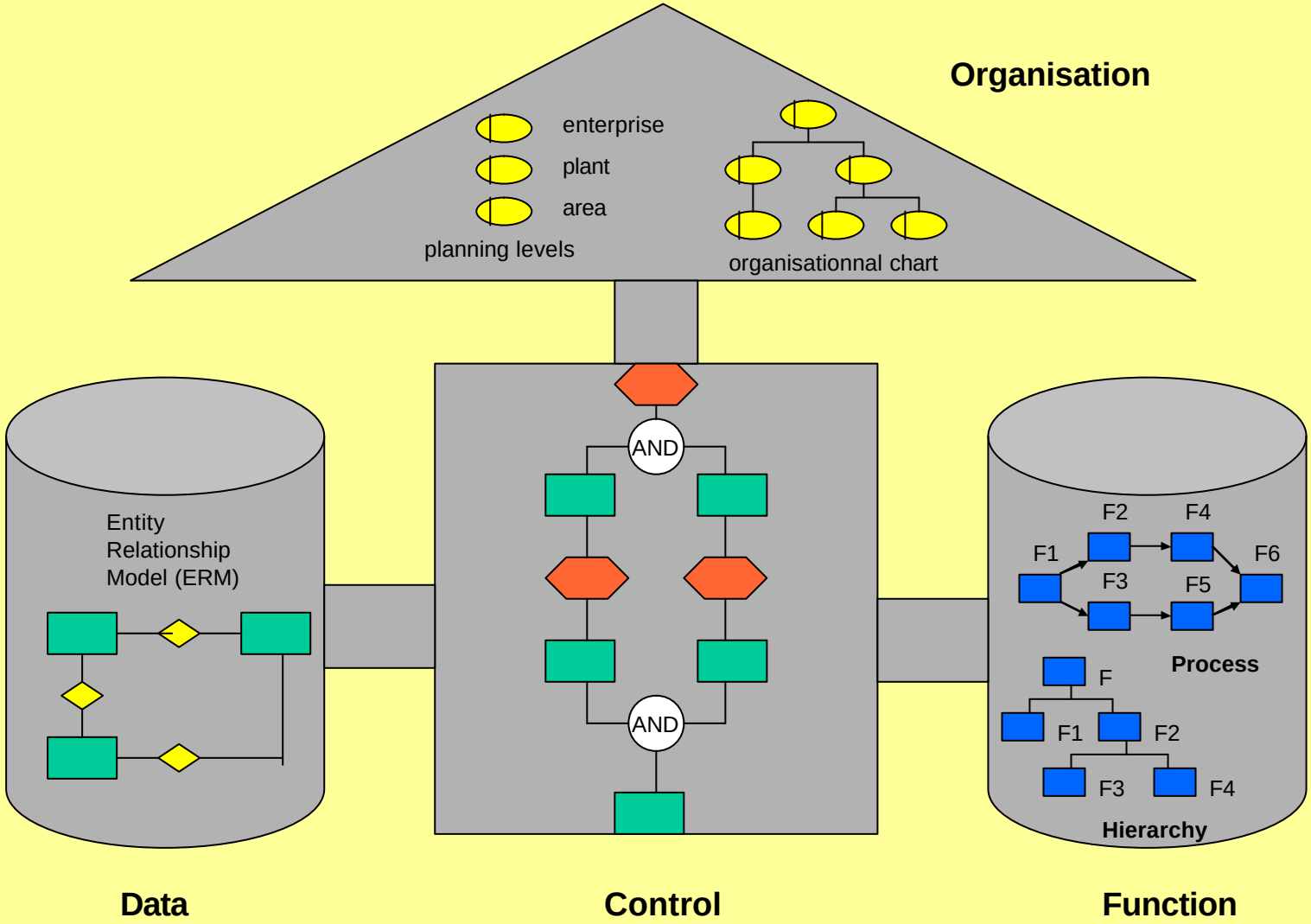
EM: Panorama of tools

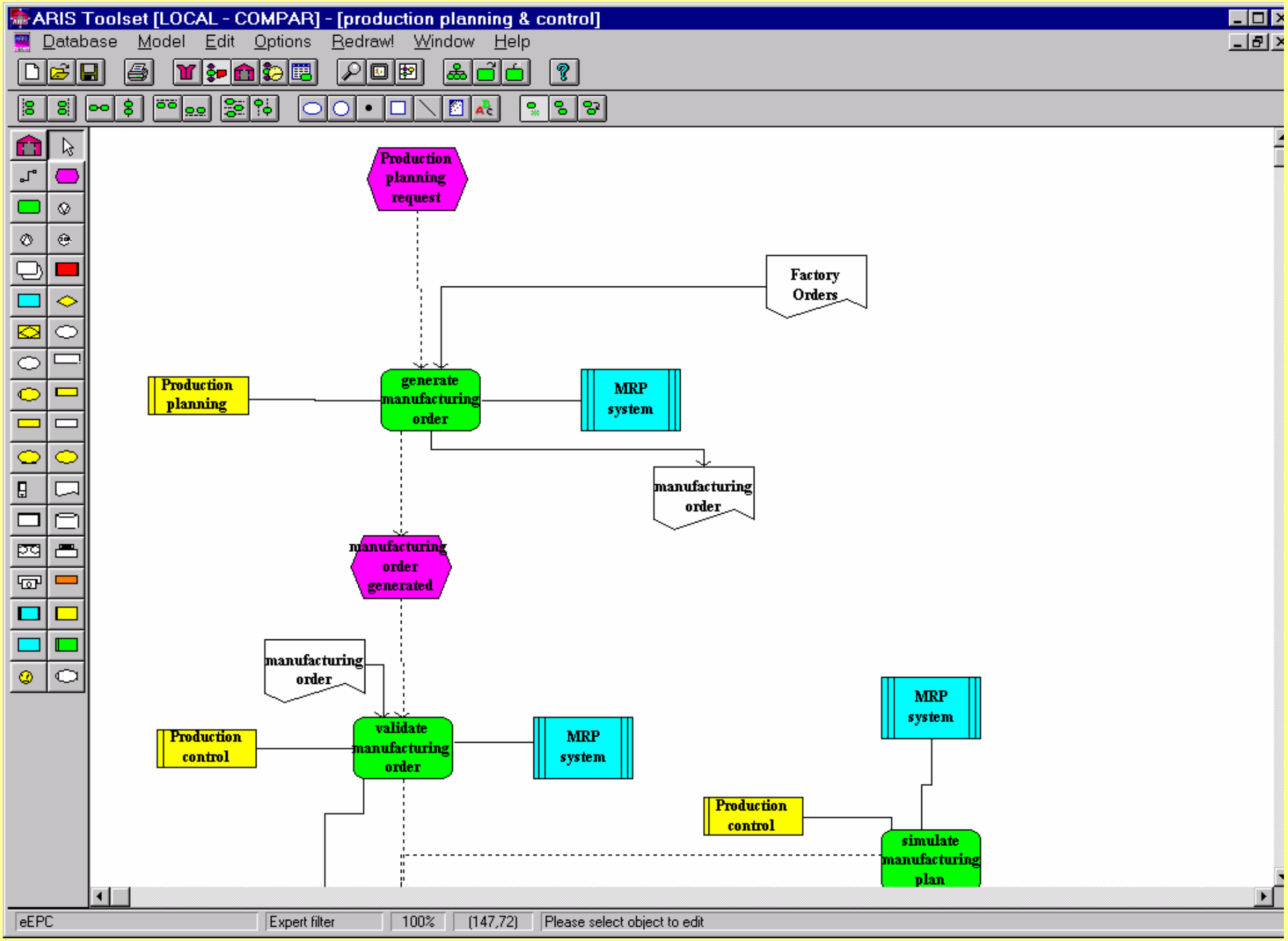
- ARIS ToolSet (IDS Scheer)
- FirstSTEP (Interfacing Technologies)
- Metis (NCR)
- *PACE* (IBE Simulation Engineering)
- MooGo/IEM (IPK Berlin)
- CimTool (RGCP)
- GraiTools 1.0 (GraiSoft)

ARIS ToolSet

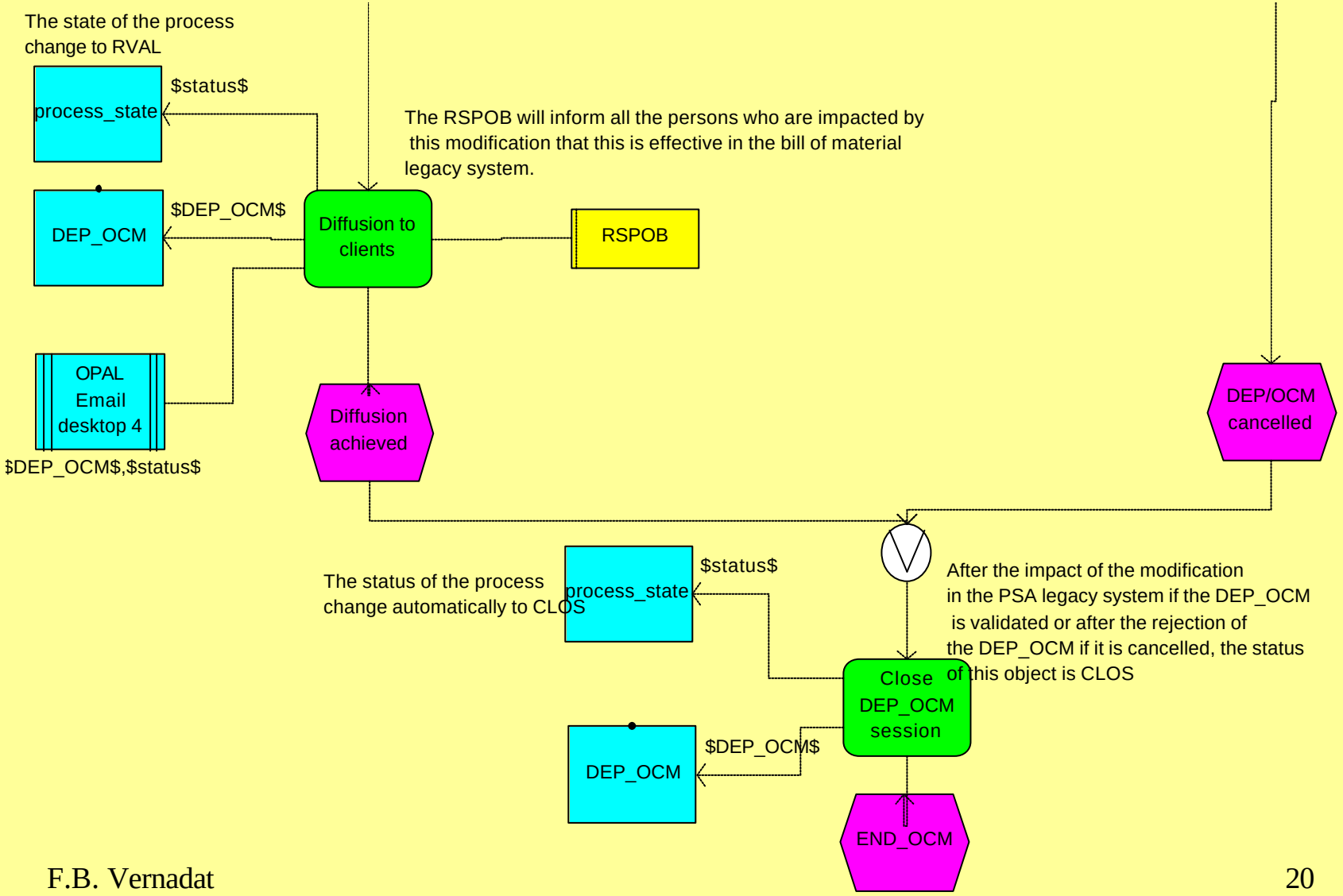
- IDS Scheer, Germany (1993)
- Number 1 in sales worldwide
- Four integrated modelling views
- Event Process Chain (EPC model)
- Business process analysis orientation
- Software engineering orientation
- Limited simulation capabilities
- Runs on Windows, Unix, Linux PCs or servers

ARIS Architecture



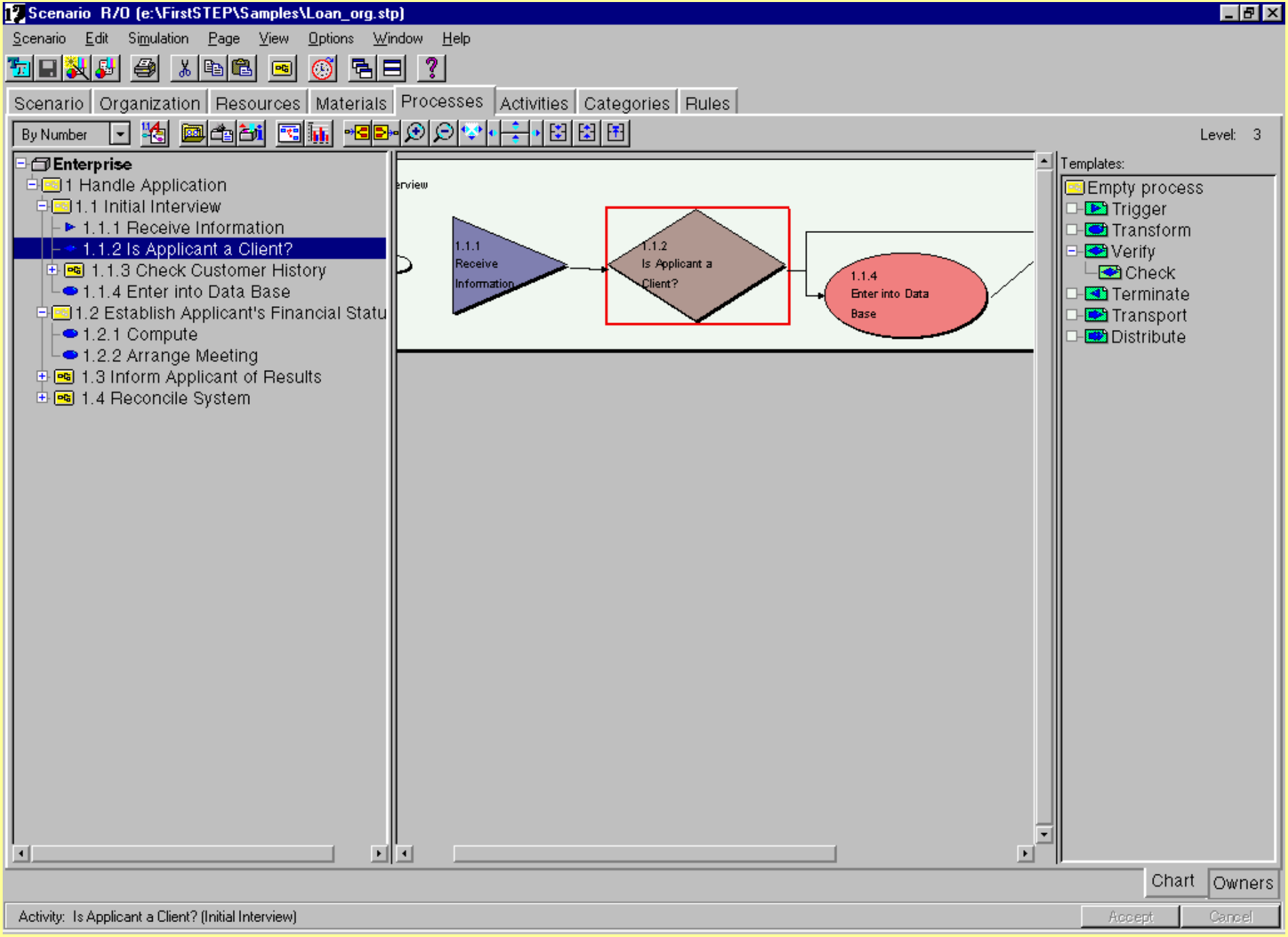

ARIS Toolset [LOCAL - COMPAR] - [production planning & control]
Database
Model
Edit
Options
Redraw!
Window
Help
Production planning request
Factory Orders
Production planning
generate manufacturing order
MRP system
manufacturing order
manufacturing order generated
manufacturing order
Production control
validate manufacturing order
MRP system
MRP system
Production control
simulate manufacturing plan
eEPC
Expert filter
100%
(147,72)
Please select object to edit

ARIS ToolSet

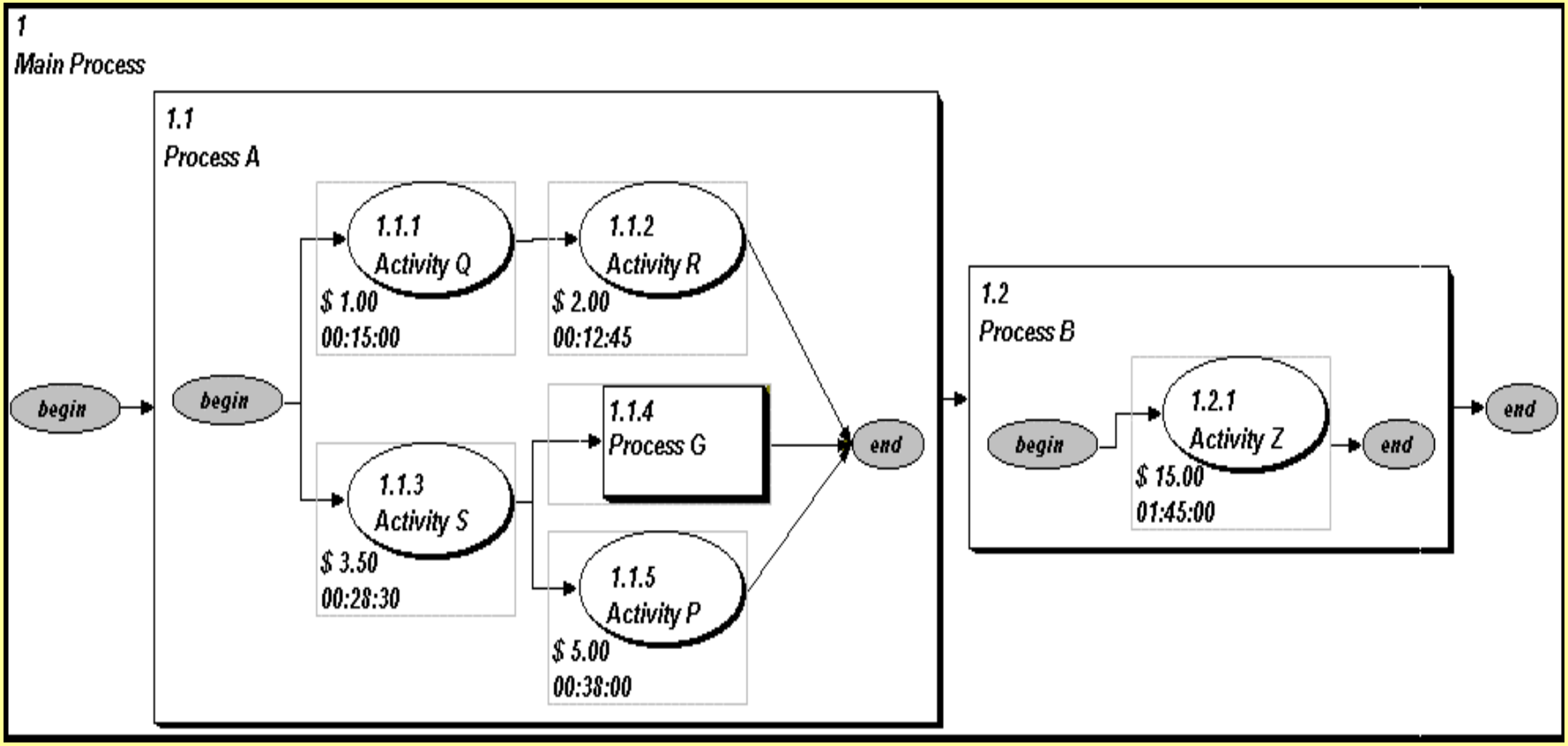


FirstSTEP

- Interfacing Technologies (Montreal, CDN)
- Kernel developed by NRC Canada, Ottawa
- Modelling and decision-support tool
- Well-suited for “What-if” scenarios by managers
- Embedded simulation capabilities
  (especially costs and resource capacities)
- Runs on Windows PCs

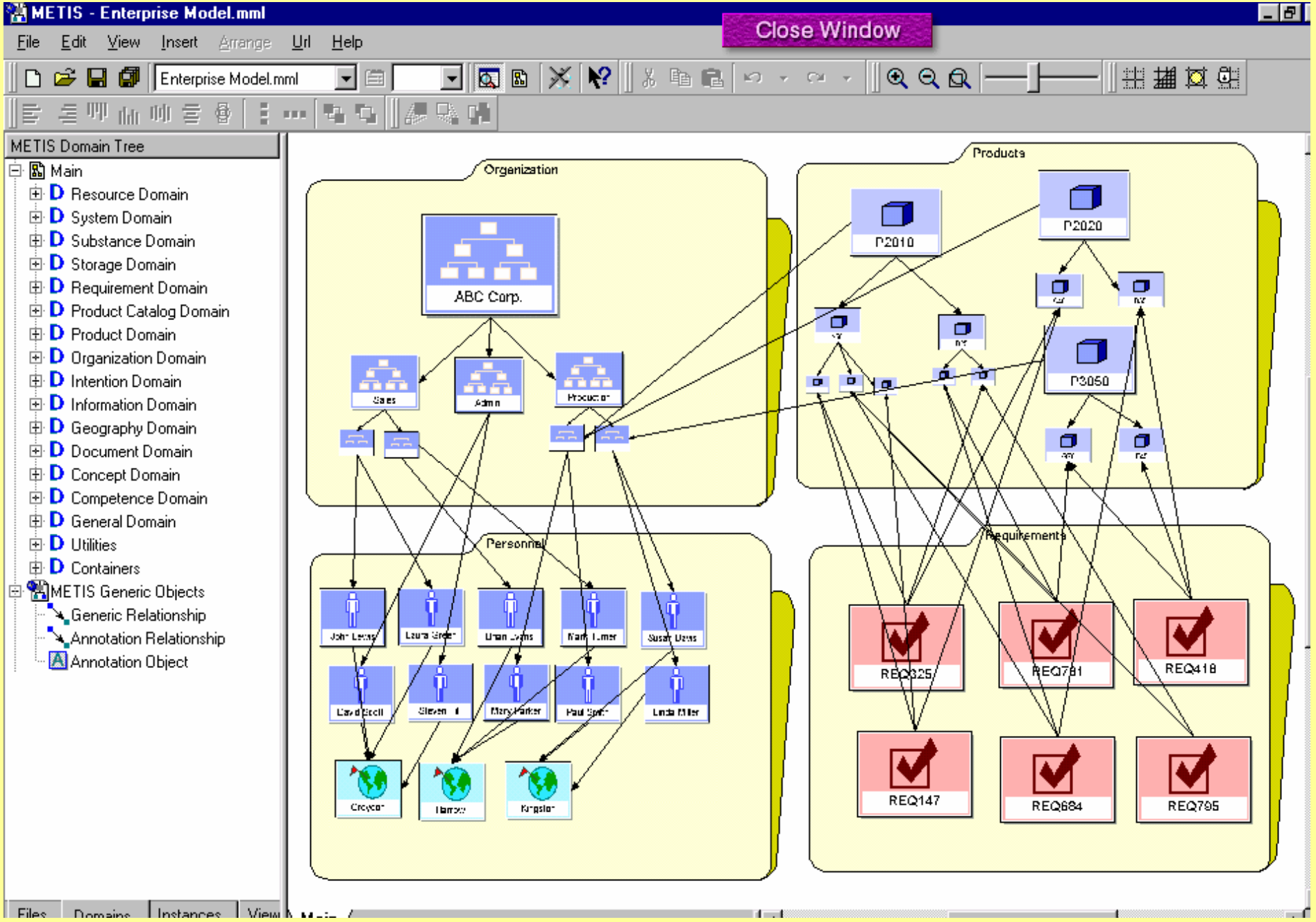

Scenario R/O (e:\FirstSTEP\Samples\Loan_org.stp)
Scenario Edit Simulation Page View Options Window Help
Scenario Organization Resources Materials Processes Activities Categories Rules
By Number
Level: 3
Enterprise
1 Handle Application
1.1 Initial Interview
1.1.1 Receive Information
1.1.2 Is Applicant a Client?
1.1.3 Check Customer History
1.1.4 Enter into Data Base
1.2 Establish Applicant's Financial Statu
1.2.1 Compute
1.2.2 Arrange Meeting
1.3 Inform Applicant of Results
1.4 Reconcile System
erview
1.1.1 Receive Information
1.1.2 Is Applicant a Client?
1.1.4 Enter into Data Base
Templates:
Empty process
Trigger
Transform
Verify
Check
Terminate
Transport
Distribute
Chart
Owners
Activity: Is Applicant a Client? (Initial Interview)
Accept
Cancel

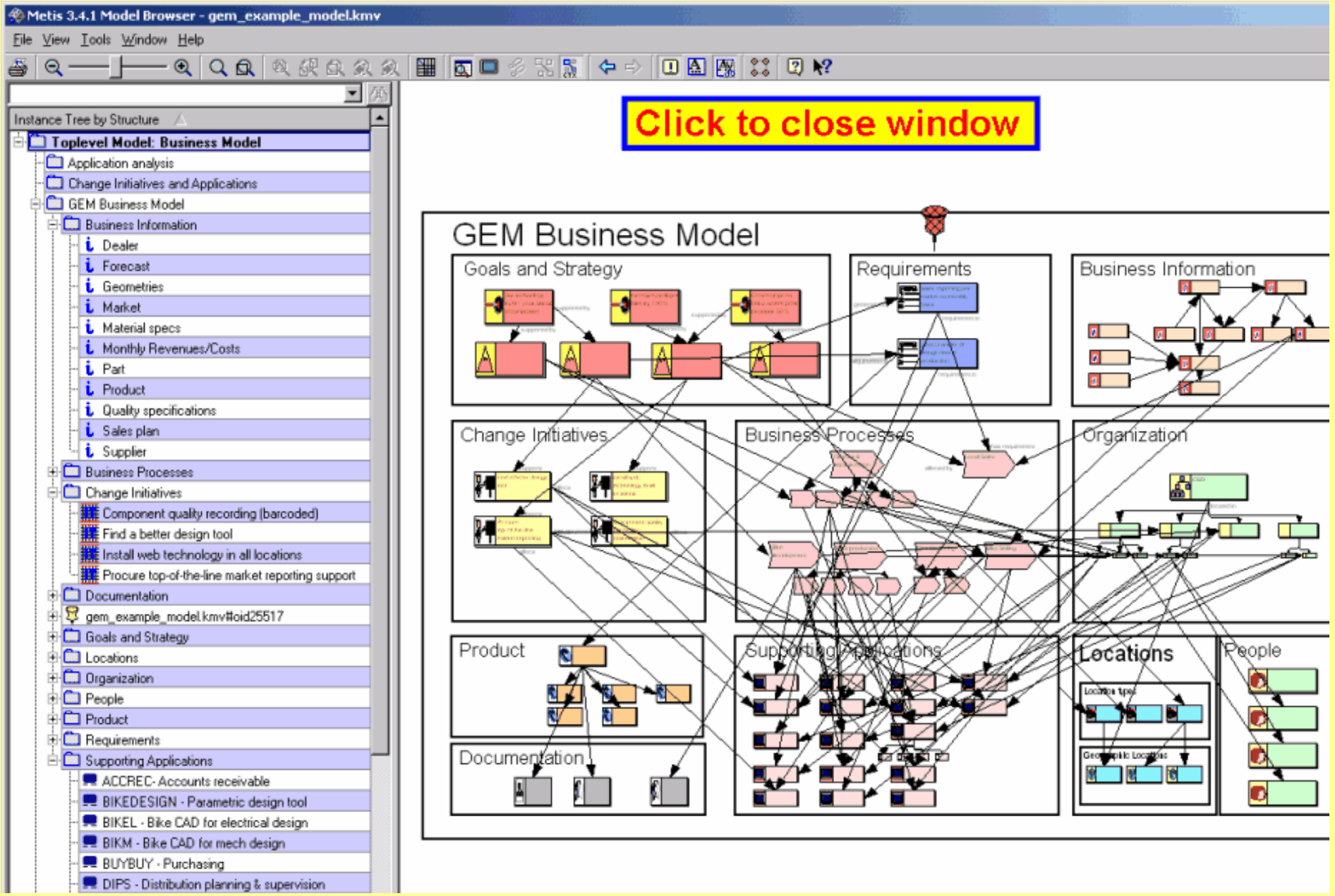
FirstSTEP


1
Main Process
1.1
Process A
1.1.1
Activity Q
$ 1.00
00:15:00
1.1.2
Activity R
$ 2.00
00:12:45
1.1.3
Activity S
$ 3.50
00:28:30
1.1.4
Process G
1.1.5
Activity P
$ 5.00
00:38:00
begin
begin
end
1.2
Process B
begin
1.2.1
Activity Z
$ 15.00
01:45:00
end
end

Metis

- Developed by NCR Inc. and Computas (Norway)
- Enterprise knowledge acquisition & vizualisation tool
- Used for Enterprise Architecture definition
- Good for organization analysis and design
- Runs on Windows

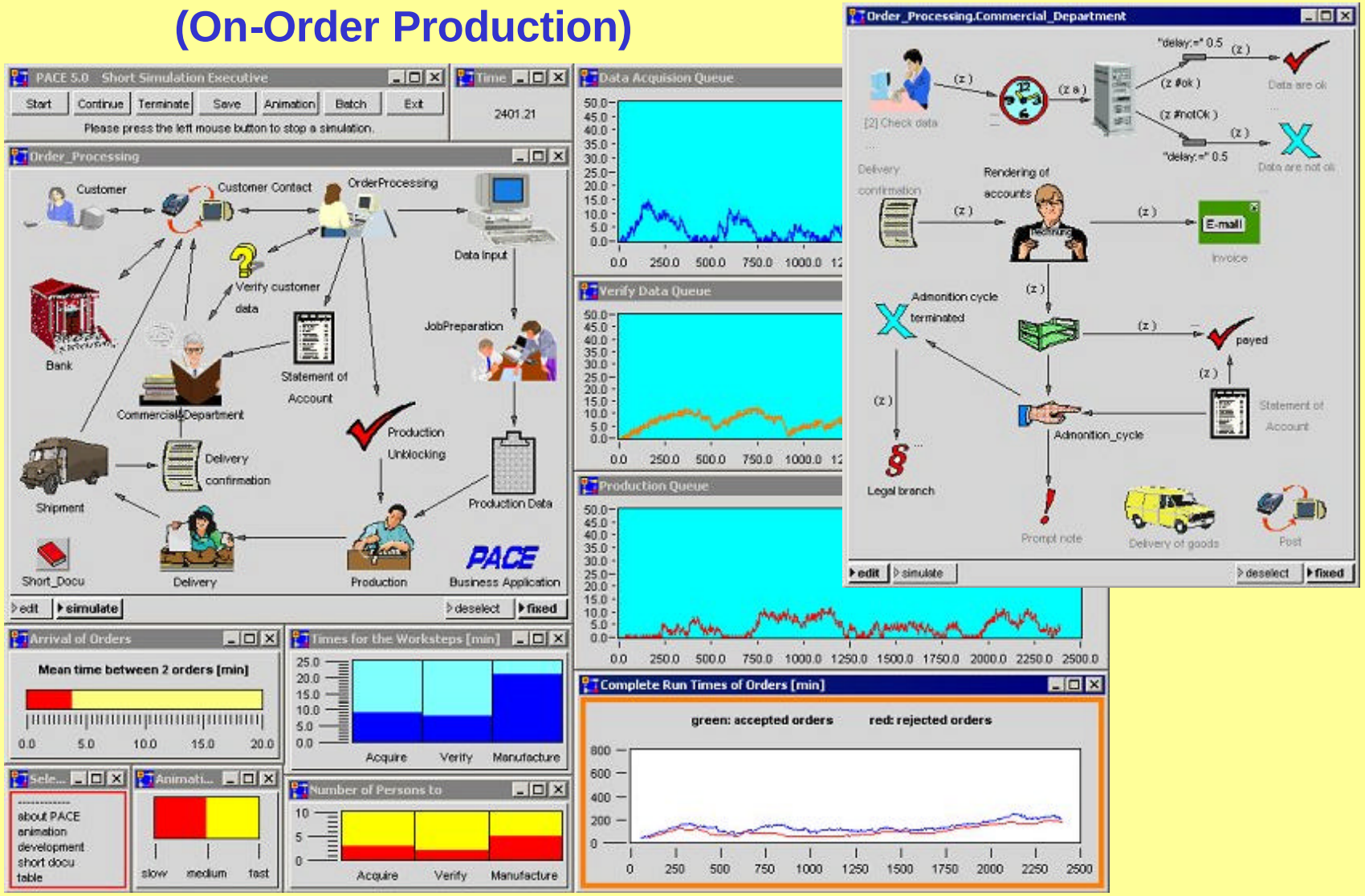

METIS - Enterprise Model.mml
Close Window
File
Edit
View
Insert
Arrange
Url
Help
Enterprise Model.mml
METIS Domain Tree
Main
Resource Domain
System Domain
Substance Domain
Storage Domain
Requirement Domain
Product Catalog Domain
Product Domain
Organization Domain
Intention Domain
Information Domain
Geography Domain
Document Domain
Concept Domain
Competence Domain
General Domain
Utilities
Containers
METIS Generic Objects
Generic Relationship
Annotation Relationship
Annotation Object
Files
Domains
Instances
Organization
ABC Corp.
Sales
Admin
Products
P2010
P2020
P3050
Personnel
Requirements
REQ325
REQ781
REQ418
REQ147
REQ684
REQ795

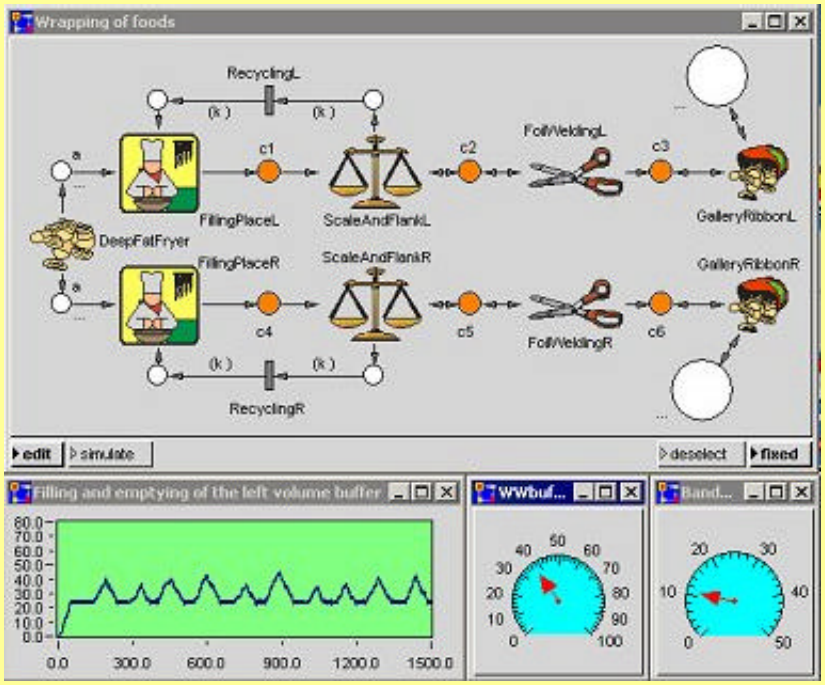

Metis 3.4.1 Model Browser - gem_example_model.kmv
File View Tools Window Help
Instance Tree by Structure
Toplevel Model: Business Model
Application analysis
Change Initiatives and Applications
GEM Business Model
Business Information
Dealer
Forecast
Geometries
Market
Material specs
Monthly Revenues/Costs
Part
Product
Quality specifications
Sales plan
Supplier
Business Processes
Change Initiatives
Component quality recording (barcoded)
Find a better design tool
Install web technology in all locations
Procure top-of-the-line market reporting support
Documentation
gem_example_model.kmv#oid25517
Goals and Strategy
Locations
Organization
People
Product
Requirements
Supporting Applications
ACCREC - Accounts receivable
BIKEDESIGN - Parametric design tool
BIKEL - Bike CAD for electrical design
BIKM - Bike CAD for mech design
BUYBUY - Purchasing
DIPS - Distribution planning & supervision
Click to close window
GEM Business Model
Goals and Strategy
Requirements
Business Information
Change Initiatives
Business Processes
Organization
Product
Supporting Applications
Locations
People
Documentation

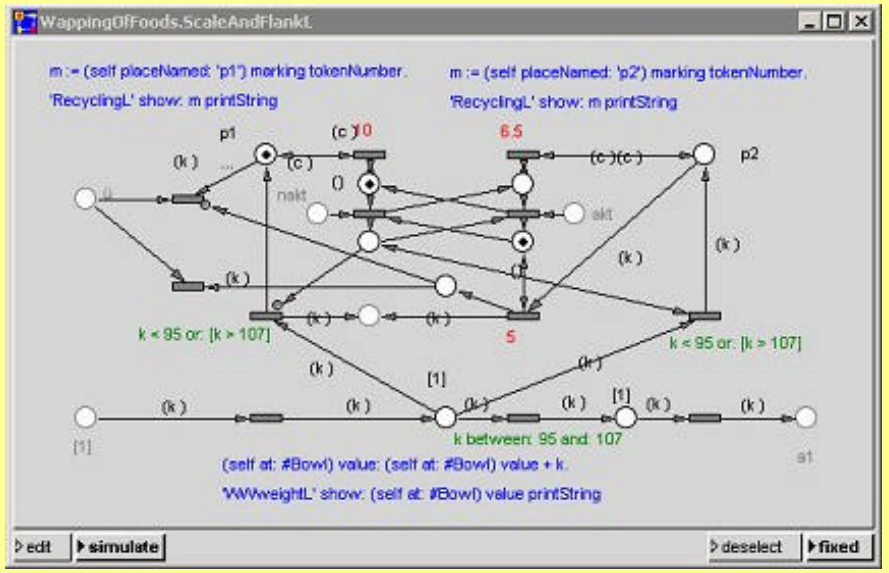
PACE

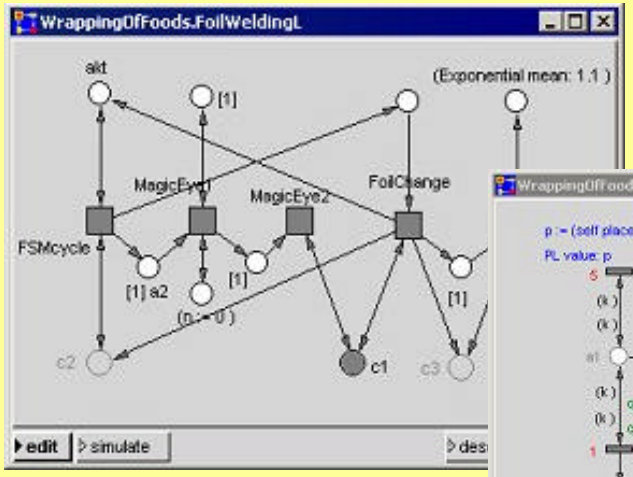
- IBE Simulation Engineering GmbH, Germany (1994)
- Number of installations worldwide > 450
- (Modelling -> Simulation -> Visualization -> Optimization)
  of technical and of business processes
- Semi-graphical modelling language MSL
  based on attributed Petri-nets
- Hierarchical net models (modular model structure)
- Many advanced features integrated like
  Fuzzy techniques, net procedures, optimization methods, empirical probability distributions, etc.

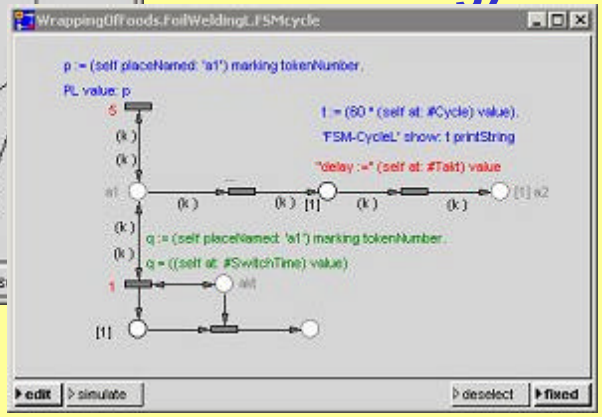
PACE: Process Modelling with Simple Application Interfaces for the Daily Use. (On-Order Production)



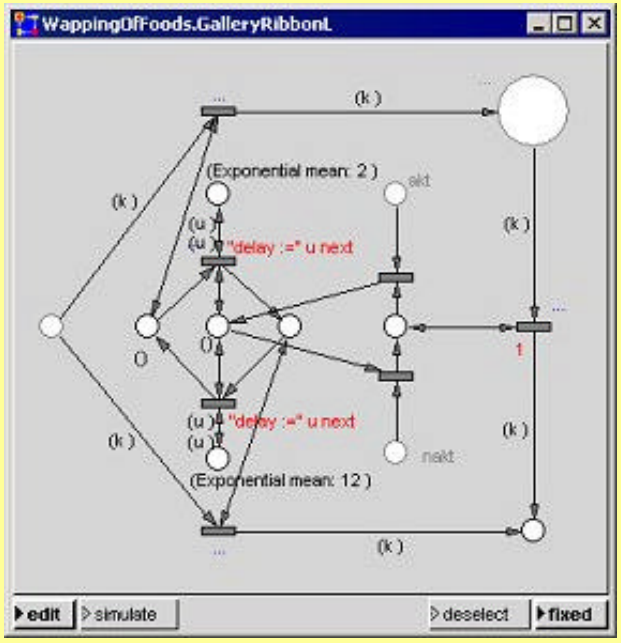








PACE Modelling Example (Food Packing)



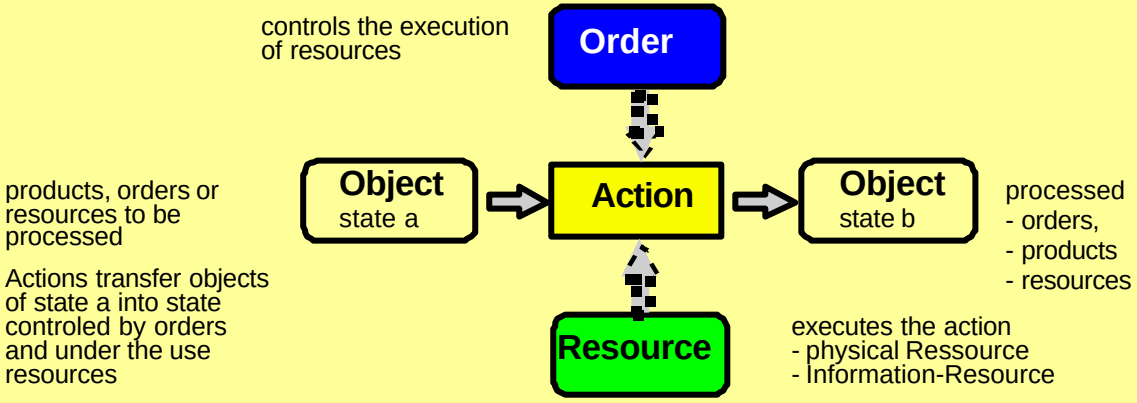
MooGo/IEM

- Commercialised by PSI, Germany
- Developed by IPK Berlin (1994)
- Based on SADT model
- Strong object orientation
- Three fundamental types of objects:

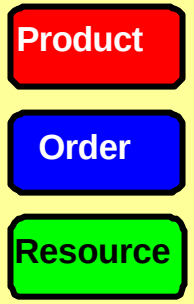
  Order, Product, Resource

MooGo/IEM

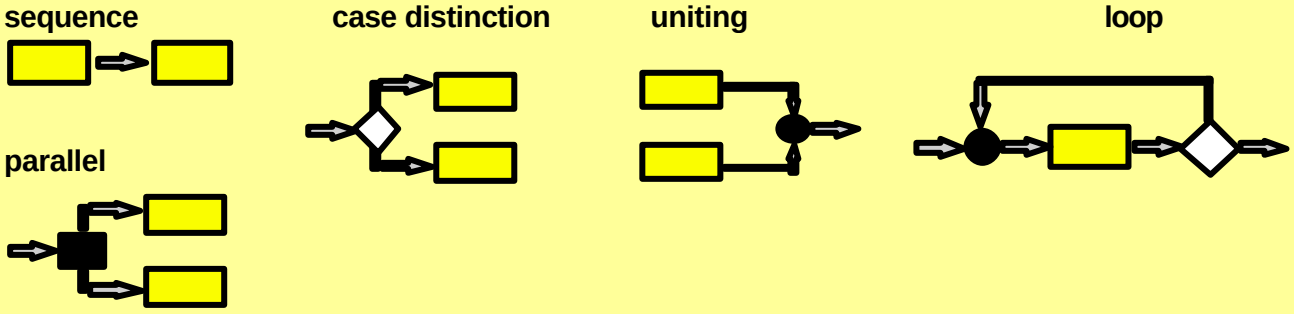
a) descriptive Features



b) modeled Objects

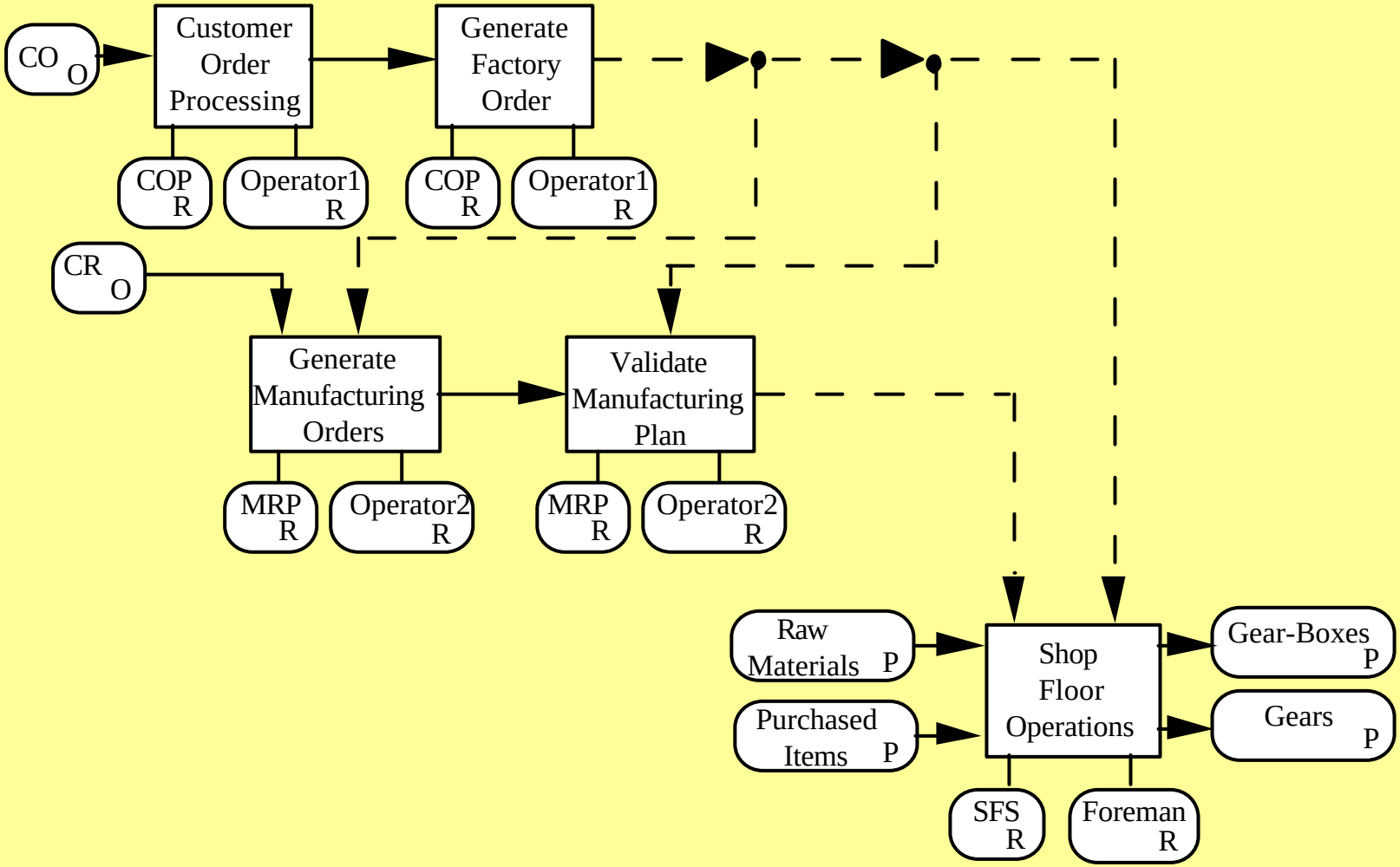


c) connecting Elements



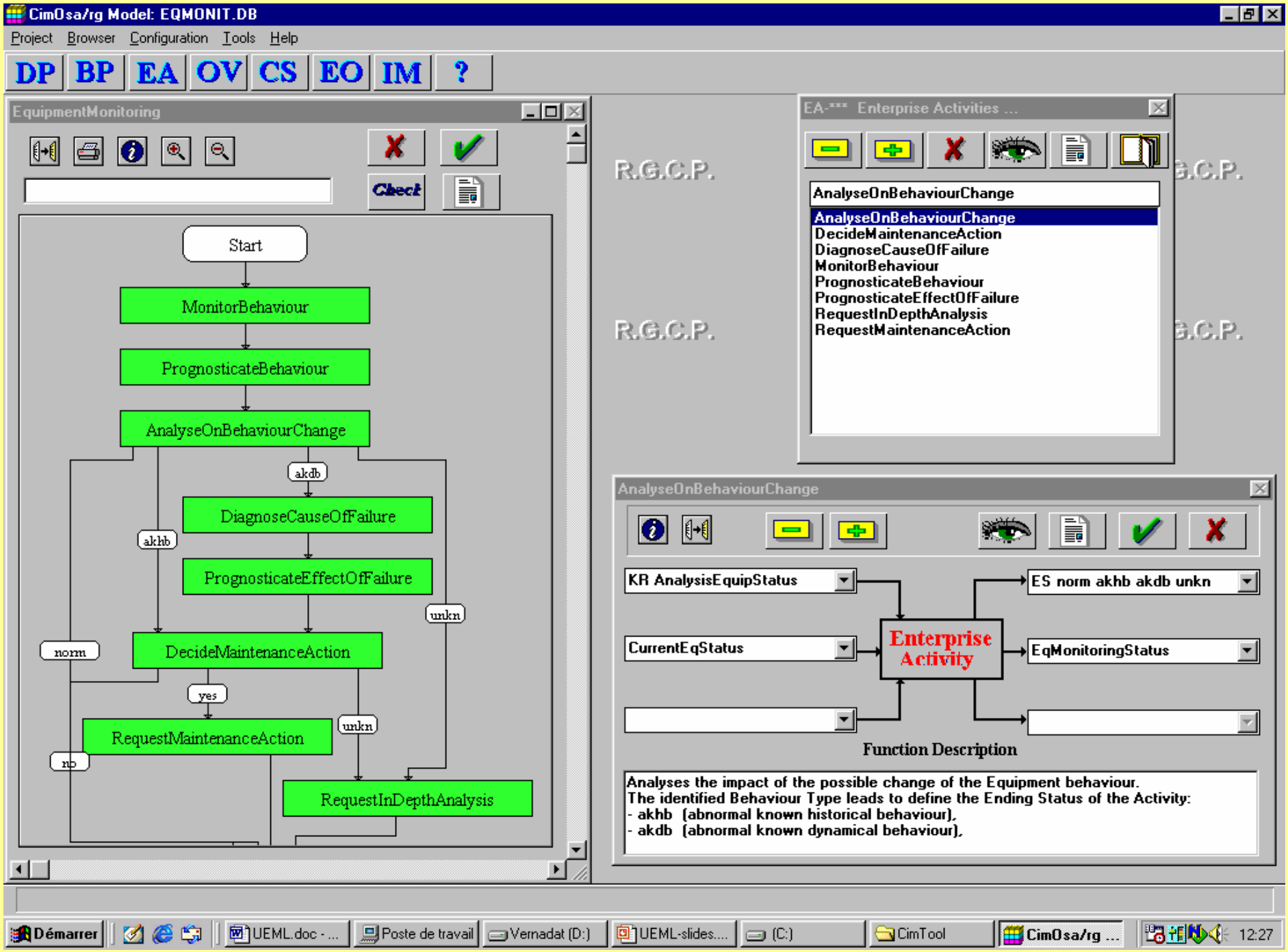
Ouelle: IPK / MO²GO

MooGo/IEM

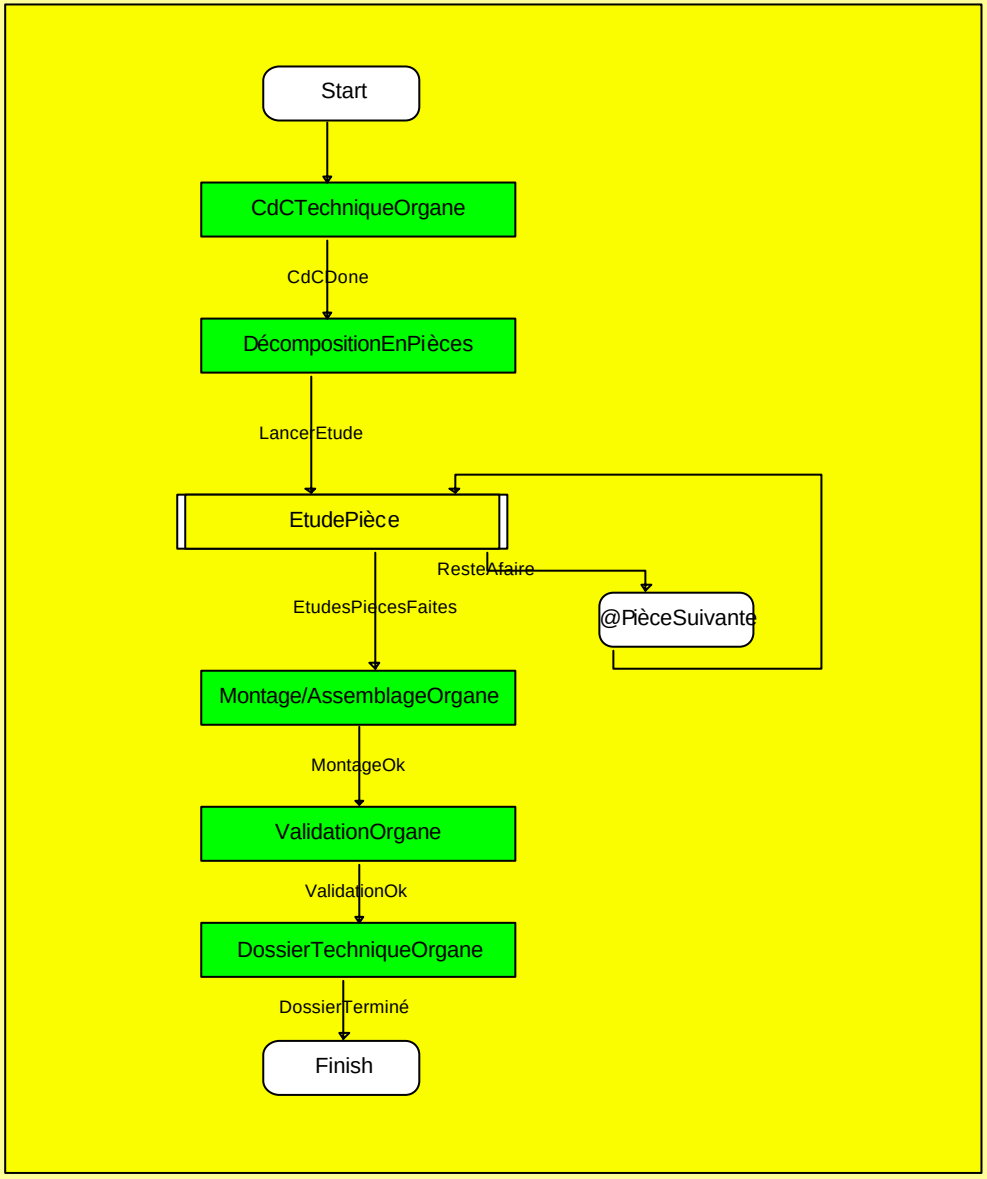

CO
O
Customer Order Processing
Generate Factory Order
COP
R
Operator1
R
COP
R
Operator1
R
CR
O
Generate Manufacturing Orders
Validate Manufacturing Plan
MRP
R
Operator2
R
MRP
R
Operator2
R
Raw Materials
P
Purchased Items
P
Shop Floor Operations
Gear-Boxes
P
Gears
P
SFS
R
Foreman
R

CimTool

- Developed by René Gaches Consultants (1995)
- Based on CIMOSA constructs
- Limited to modelling
  (function and information views mostly)
- Very easy to learn and to use
- Runs on Windows PCs only


CimOsa/rg Model: EQMONIT.DB
Project Browser Configuration Tools Help
DP BP EA OV CS EO IM ?
EquipmentMonitoring
Check
Start
MonitorBehaviour
PrognosticateBehaviour
AnalyseOnBehaviourChange
akdb
DiagnoseCauseOfFailure
akhb
PrognosticateEffectOfFailure
unkn
norm
DecideMaintenanceAction
yes
unkn
RequestMaintenanceAction
nb
RequestInDepthAnalysis
EA-*** Enterprise Activities ...
AnalyseOnBehaviourChange
AnalyseOnBehaviourChange
DecideMaintenanceAction
DiagnoseCauseOfFailure
MonitorBehaviour
PrognosticateBehaviour
PrognosticateEffectOfFailure
RequestInDepthAnalysis
RequestMaintenanceAction
R.G.C.P.
R.G.C.P.
AnalyseOnBehaviourChange
KR AnalysisEquipStatus
CurrentEqStatus
Enterprise Activity
ES norm akhb akdb unkn
EqMonitoringStatus
Function Description
Analyses the impact of the possible change of the Equipment behaviour.
The identified Behaviour Type leads to define the Ending Status of the Activity:
- akhb (abnormal known historical behaviour),
- akdb (abnormal known dynamical behaviour),
Démarrer
UEML.doc - ...
Poste de travail
Vernadat (D:)
UEML-slides....
(C:)
CimTool
CimOsa/rg ...
12:27

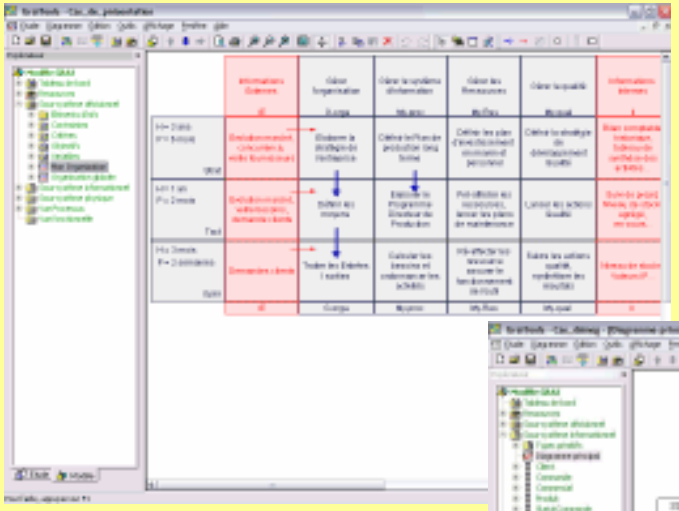
CimTool


Start
CdCTechniqueOrgane
CdCDone
DécompositionEnPièces
LancerEtude
EtudePièce
ResteAfaire
@PièceSuivante
EtudesPiecesFaites
Montage/AssemblageOrgane
MontageOk
ValidationOrgane
ValidationOk
DossierTechniqueOrgane
DossierTerminé
Finish

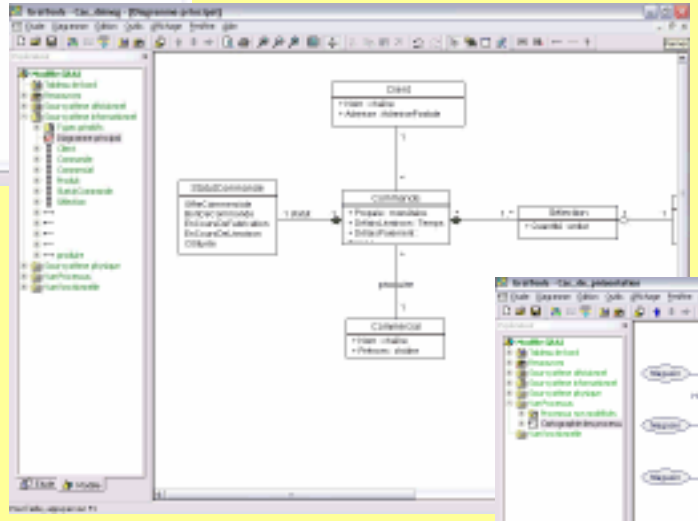
GraiTools 1.0

- Developed by GraiSoft, Bordeaux (2003)
- Specificity: based on the GRAI method
- Strength: decision centre analysis
- Complete environment for BPM, enterprise modelling and project monitoring
- Performance indicator deployment
- Runs on PCs

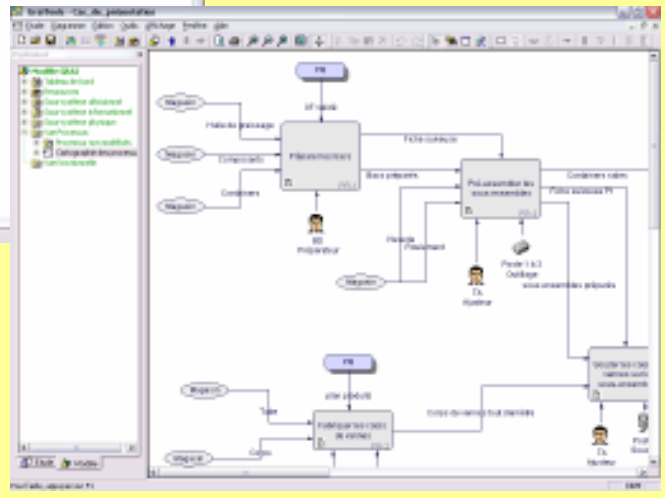
GraiTools 1.0



GRAI grid



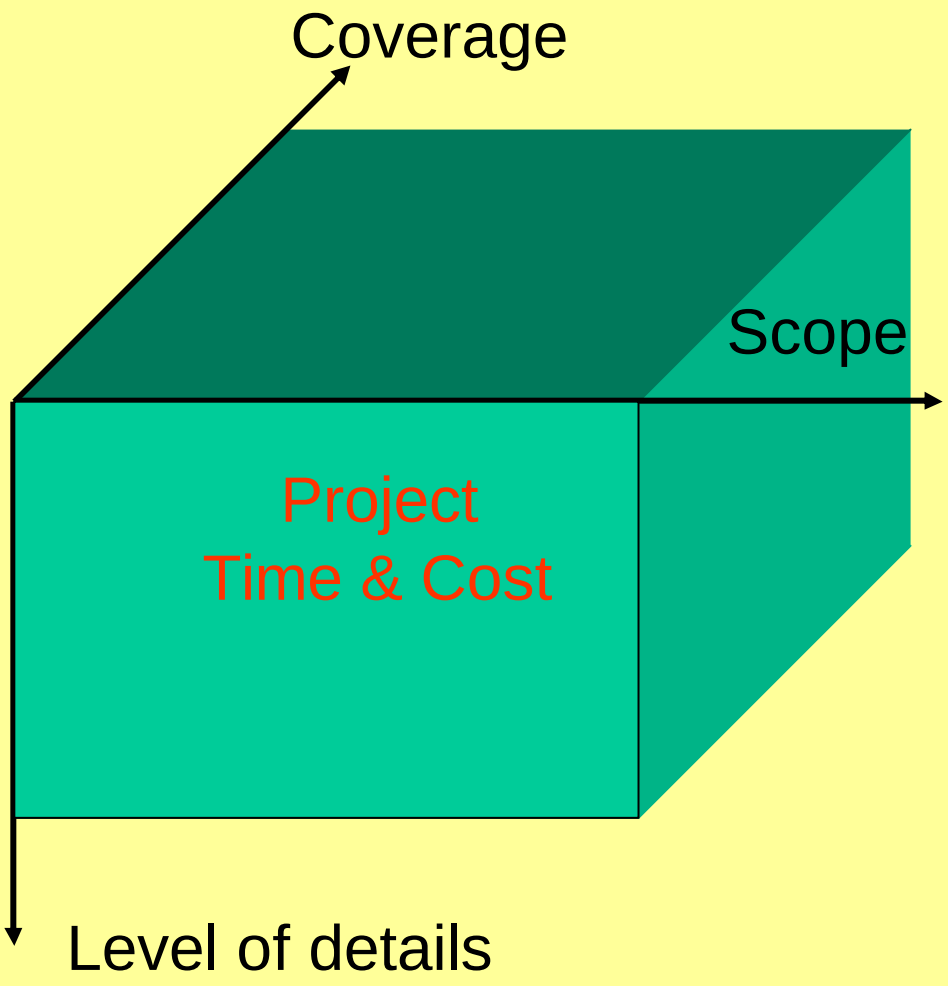
Decision centres



Business processes

Characteristics of Enterprise Models

- Scope
- Coverage
- Level of details
- Competency
- Completeness
- Consistency



Part II

Enterprise Modelling Constructs

Enterprise Modelling Constructs

Definition:

A construct is a primitive component (with syntax and semantics) of a modelling language

It is subject to rules for combining/putting together constructs to build models of any size

Appearances:

- Graphical notation (e.g. SADT/IDEF0, IDEF3, ER models)
- Template / object class notation (e.g. CIMOSA, IEM)
- Formal notation (e.g. TOVE, UEML)

ARIS Toolset Language

Event-Process Chains (EPC)

- **Event**
- **Function**
- **Information Object**
  - **Entity**
  - **Relationship**
- **Organisation Unit**
- **Employee**
- **IT-Resource**

CIMOSA Language

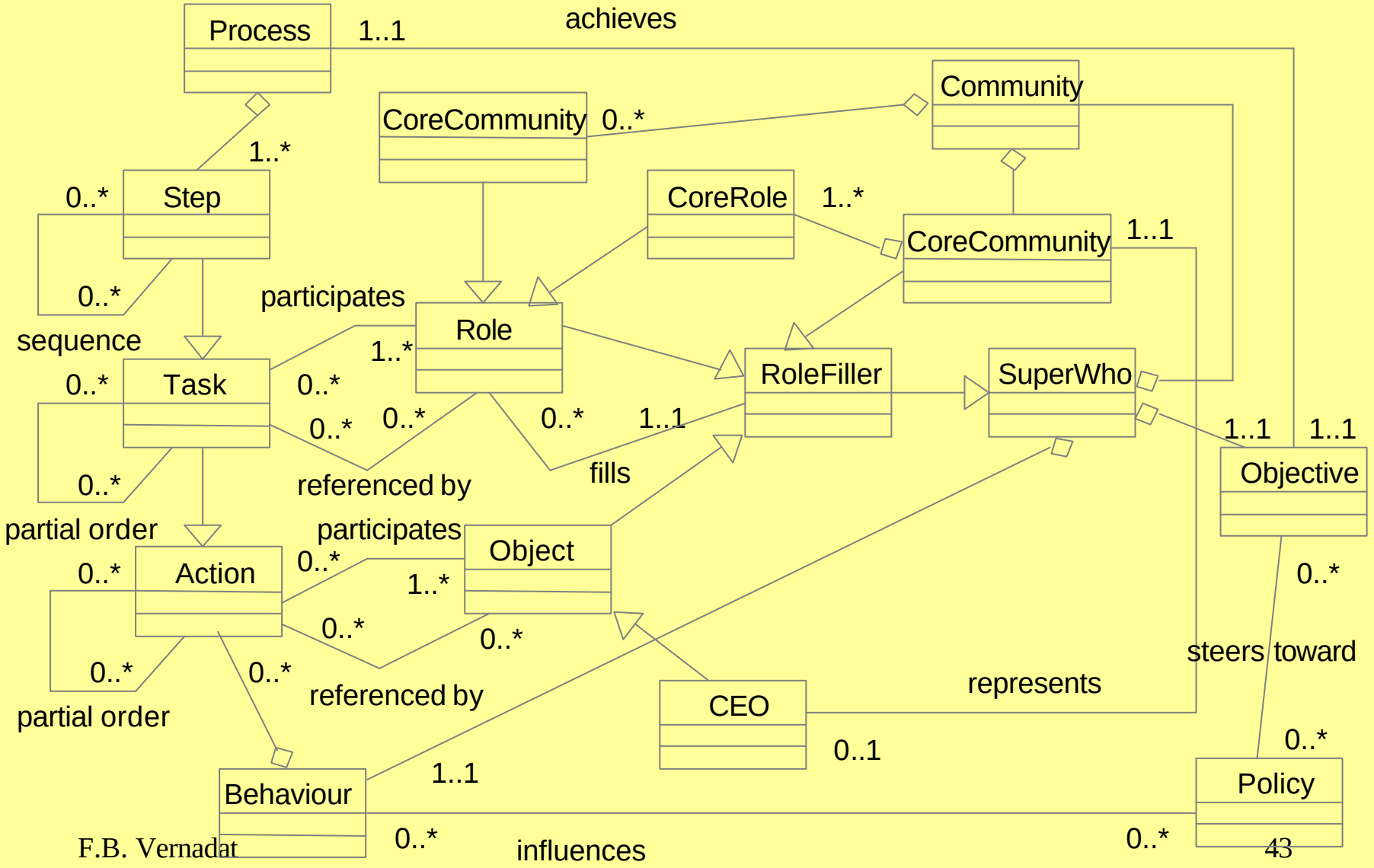
CIMOSA

Event-driven process-based language

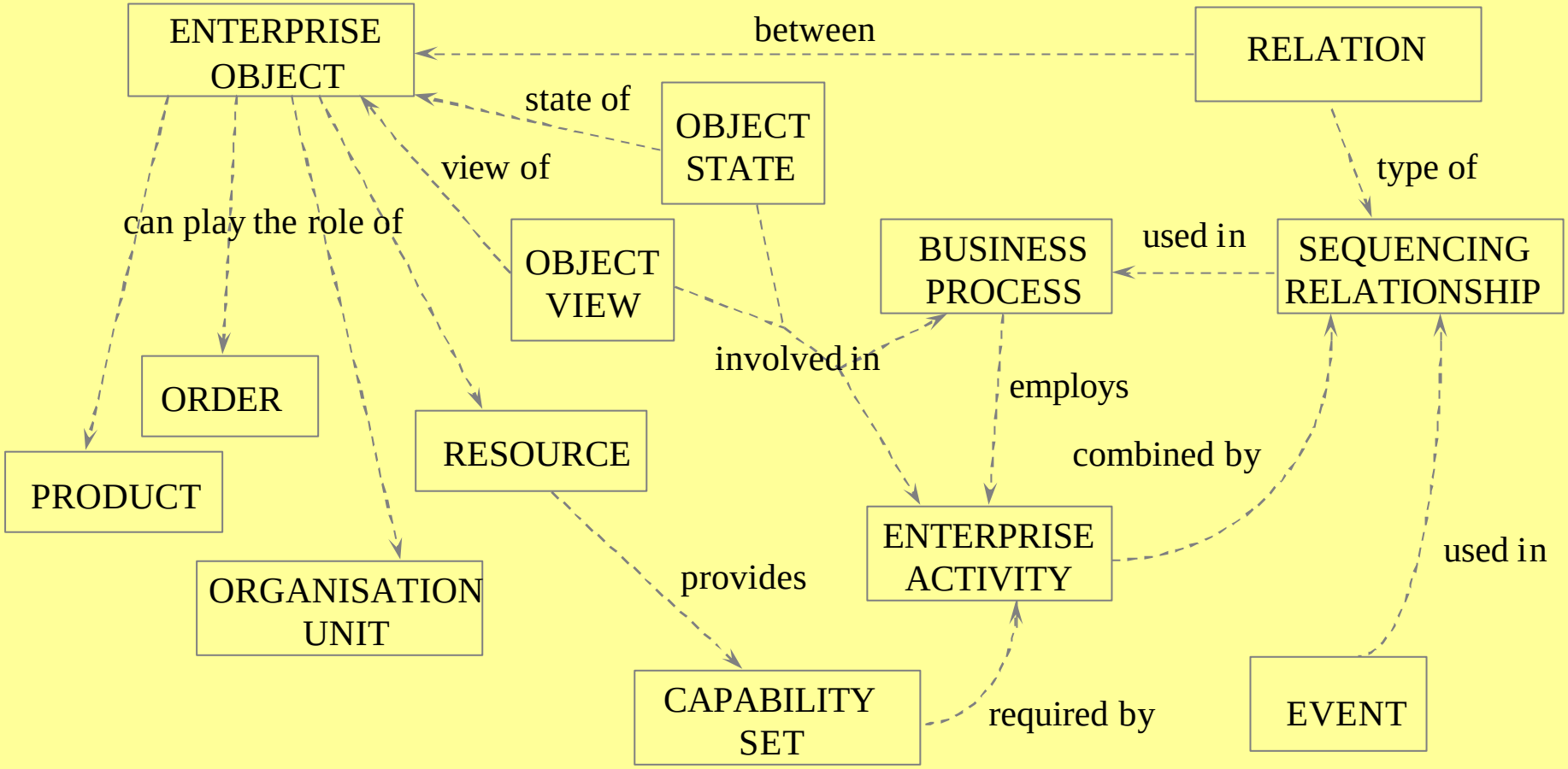
- Process/Operation/Agent relationship
- Task/Capability-Competency/Agent relationship

Event	Ev = (Eid, source, P-list, Predicate, Date)
Process	P = (Pid, ?P, ?P, ?P, ES)
	?P = {WHEN (condition) DO action}
Activity	A = (Aid, FI, CI, RI, FO, CO, RO, CS, ?A, ES)
	CS= set of capabilities/competencies
Resource	R = (Rid, CS, FO-list, components)
Object View	OV = (Ovid, O-list, Properties)
Enterprise Object	EO = (Oid, Isa, Part-of, Properties)
Organisation unit	
	UO = (UOid, tasks, responsibilities, authorities)
Organisation cell	
	EO = (Eoid, manager, responsibilities, authorities, components)

ODP Metamodel



CEN EN 12204 / prEN-ISO 19440



Comité Européen de Normalisation, Bruxelles

Enterprise Modelling paradigm

Three essential concepts: *process*, *agent* and *operation*

A process is a partially ordered sequence of steps to achieve a goal (or end-result)

Processes = What has to be done

Operations = Primitive actions (required or provided)

Agents = The Doers (those who carry out the processes)

Three companion concepts: *event*, *activity* and *object view*

Events = Process triggers (happenings requiring actions)

Activities = Process steps

Object views = Processed items (objects involved in processes)

Enterprise Modelling paradigm

Three types of processes:

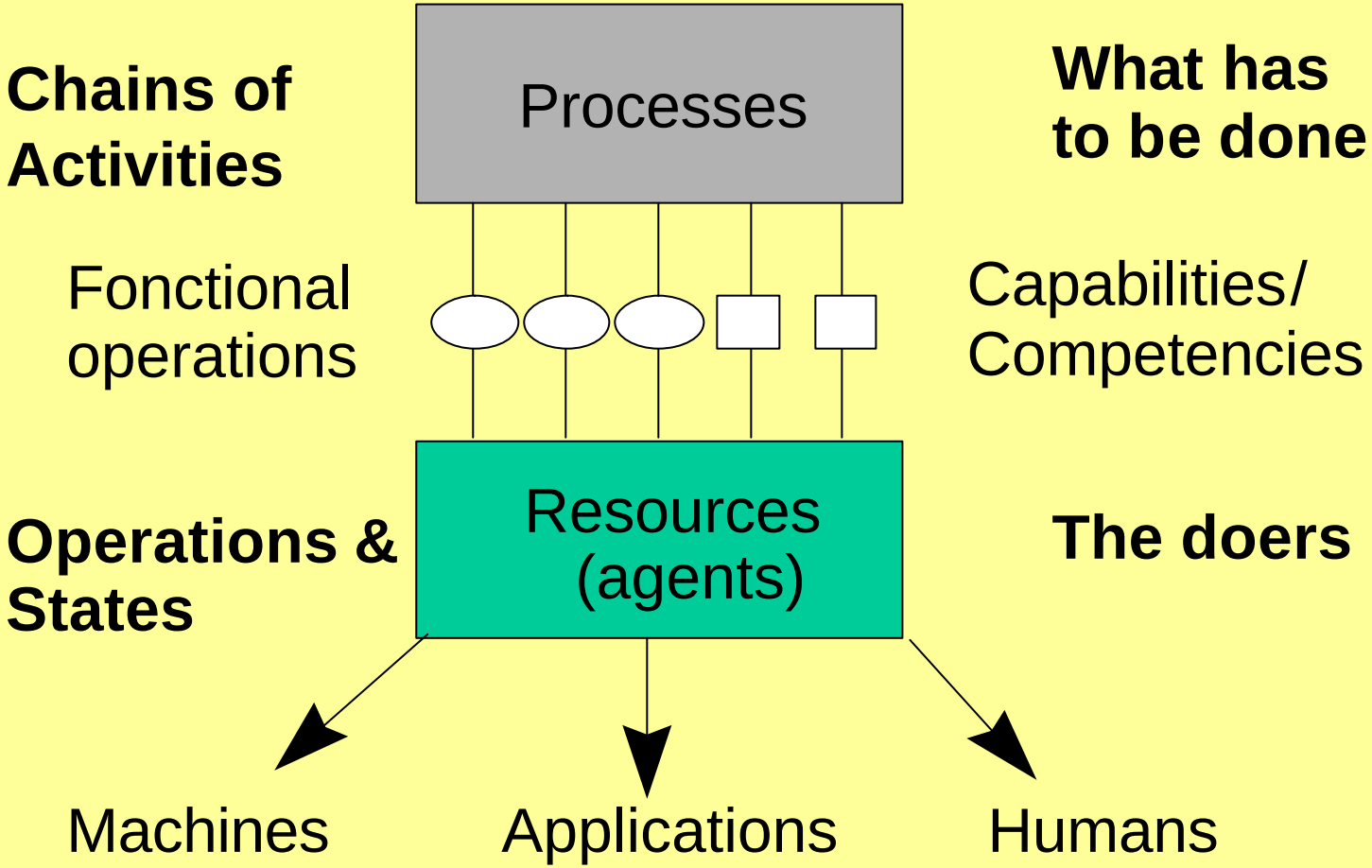
- Well-structured processes (behaviour is deterministic)
- Semi-structured processes (behaviour is partially known)
- Ill-structured processes (behaviour is not predictable)

Three types of agents:

- Machines (devices with a controller)
- Applications (IT systems)
- Humans

Enterprise Modelling paradigm

Principle of separation of processes and resources



EM Constructs: Information View

- **Enterprise Object:** Any entity (physical object or information) in the enterprise subject to be referenced in enterprise activities or in the organisation structure

  Ex.: orders, products, resources, invoices, process plans…

- **Formalisation:**

  $EO = ?EOid, Isa_{EO}, Partof_{EO}, Prop_{EO}?$

  EOid enterprise object name, Isa_{EO} generalisation/specialisation relationship of *EO*, $Partof_{EO}$ aggregation relationship of *EO*, $Prop_{EO}$ list of properties (or attributes) of *EO*. Isa_{EO} and $Partof_{EO}$ define semantic relationships of *EO* with other enterprise objects

EM Constructs: Information View

- **Object View:** Is a manifestation (state and embodiement) of one or more enterprise objects

  Ex.: Let Purchase_Order (PO) be an enterprise object, then Blank_PO, Filled_In_PO, Signed_PO are object views on PO

- **Formalisation:**

  $OV = ?OVid, EO_{OV}, Prop_{OV}?$

  OVid object view name, EO_{OV} list of enterprise objects from which the object view is derived, $Prop_{OV}$ list of properties of the object view (derived from list of properties of objects in EO_{OV})

EM Constructs: Function View

- **Event**: represents a change of state in the system
  - Solicited event: scheduled order, clock-time = 5:00 pm, etc.
  - Unsolicited event: machine break-down, alarm, error message

- **Formalisation:**

  $E = ?Eid, q, OV, t?$

  Eid agent name, *q* predicate (state change condition), *OV* object view attached to event, and *t* time-stamp for *E*

EM Constructs: Function View

- **Process:** Processes describe the enterprise behaviour in terms of flows of control (i.e. sequences of steps)
  - Core processes (or Domain processes)
  - Business processes

- **Formalisation:**

  $P = ?Pid, ?_P, ?_P, ?_P, ES_P?$

  Pid process name, $?_P$ alphabet or set of steps (i.e. activities) of P, $?_P$ set of triggering conditions c of P ($?_P = \{c / (c ?\ \ P)\}$), $?_P$ set of behavioural rules of process behaviour and ES_P finite set of ending statuses of P

EM Constructs: Function View

Process Behaviour: set of behavioural rules of the ECA form:

WHEN (condition) DO action

- Process triggering rules:

a) to start a process by means of one or more events:

WHEN (START WITH event-i AND event-j) DO EF1

b) to call a sub-process:

WHEN (START) DO EF1

- Forced sequential rules: EF2 must follow EF1 (whatever its ending status ES(EF1) is – “any” is a reserved word):

WHEN (ES(EF1) = any) DO EF2

EM Constructs: Function View

Process Behaviour (cont'd):

- Conditional sequential rules (or branching rules)**:**

Branching conditions in the flow of control

WHEN (ES(EF1) = end_stat_1) DO EF2
WHEN (ES(EF1) = end_stat_2) DO EF3
WHEN (ES(EF1) = end_stat_3) DO EF4

- Spawning rules:

Parallel execution of enterprise process steps (& operator)

(a) Asynchronous spawning:

WHEN (ES(EF1) = value) DO EF2 & EF3 & EF4

(b) Synchronous spawning: (use of SYNC reserved word)

WHEN (ES(EF1) = value) DO SYNC (EF2 & EF3 & EF4)

EM Constructs: Function View

Process Behaviour (cont'd):

- Rendez-vous rules (or flow synchronisation)**:**

To synchronise convergent control flows

WHEN (ES(EF2) = value_2 AND ES(EF3) = value_3 AND ES(EF4) = value_4) DO EF5

- Loop rules (constructed with conditional rules)**:**

WHEN (ES(EF1) = loop_value) DO EF1

WHEN (ES(EF1) = exit_value) DO EF2

- Process completion rules:

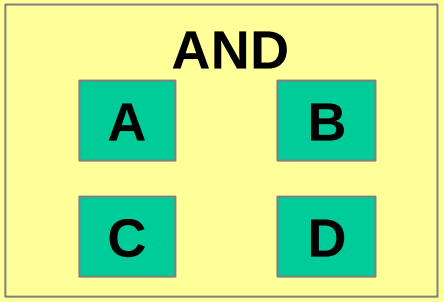
To mark the end of the control flow (use of FINISH reserved word)

WHEN (ES(EF2) = end_stat_x AND ES(EF2) = end_stat_y) DO FINISH

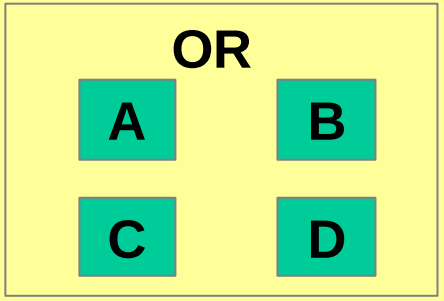
EM Constructs: Function View

Process Behaviour for ill-structured processes:

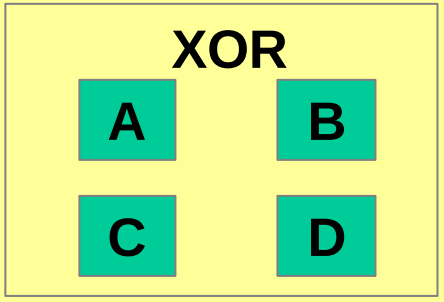
AND-case:
(do all in whatever order)



OR-case:
(do some in whatever order)



XOR-case:
(do only one on your choice)



EM Constructs: Function View

- **Activity:** a process step that is the locus of action

  (i.e. it transforms inputs into outputs using resources and time)

- **Formalisation:**

  $A = ?Aid, FI_A, FO_A, CI_A, CO_A, RI_A, RO_A, ?_A, ES_A, Cap_A, Dmin, Davg, Dmax?$

  Aid activity name, FI_A, FO_A, CI_A, CO_A, RI_A, RO_A function input, function output, control input, control output, resource input and resource output of *A*, respectively (with FI_A ? FO_A ? CI_A ? ? ; FI_A ? CI_A = ? ; FI_A, FO_A, CI_A, RO_A ? 2^{OV}; RI_A ? 2^R; CO_A ? 2^E), $?_A$ activity behaviour (algorithm or script) such that $?_A$ (FI_A, CI_A, RI_A) = (FO_A, CO_A, RO_A), ES_A finite set of ending statuses, Cap_A set of capabilities required by *A*

  $ES: A$? ? $_{A\ ?\ \mathbf{A}}$ ES_A such that ES (A) ? ES_A

  Dmin, *Davg*, *Dmax*: minimum, average, maximum duration of *A*

EM Constructs: Function View

- **Operation:** elementary action – lowest level of functional granularity
  - Used in activity behaviour ($?_A$)
  - Can be provided by several categories of agents
  - But is provided by only one agent at run-time
  - Ex.: move a part, drill a hole, update a database…

- **Formalisation:**

  agent.*operation-identifier* (IN *parameters*, OUT *parameters*)

EM Constructs: Resource View

- **Resource:** Enterprise object used as a support in the execution of an activity
  - **Agent** of **Functional Entity**: active resource (has autonomy)
  - **Component**: passive resource (e.g. a tool, a cart, etc.)

- **Formalisation** (of agents)**:**

  $R = ?Rid, OV_R, Cap_R, FO_R, f_R?$

  Rid agent name, OV_R object view describing properties of resource *R*, Cap_R finite set of capabilities offered by resources *R*, FO_R set of functional operations of *R* and f_R availability calendar of *R*

EM Constructs: Resource View

- **Capability Set:** defines a set of capabilities (i.e. technical characteristics) for technical agents or a set of competencies (i.e. skills) for human agents
  - Apply to activities (required capabilities/competencies)
  - Apply to agents (provided capabilities/competencies)

- **Categories:**

Capabilities
- Function-related
- Object-related
- Capacity-related
- Performance-related

Competencies/skills
- Knowledge
- Know-how
- Human behaviour & character traits

EM Constructs: Organisation View

Models the organisational structure of the enterprise.

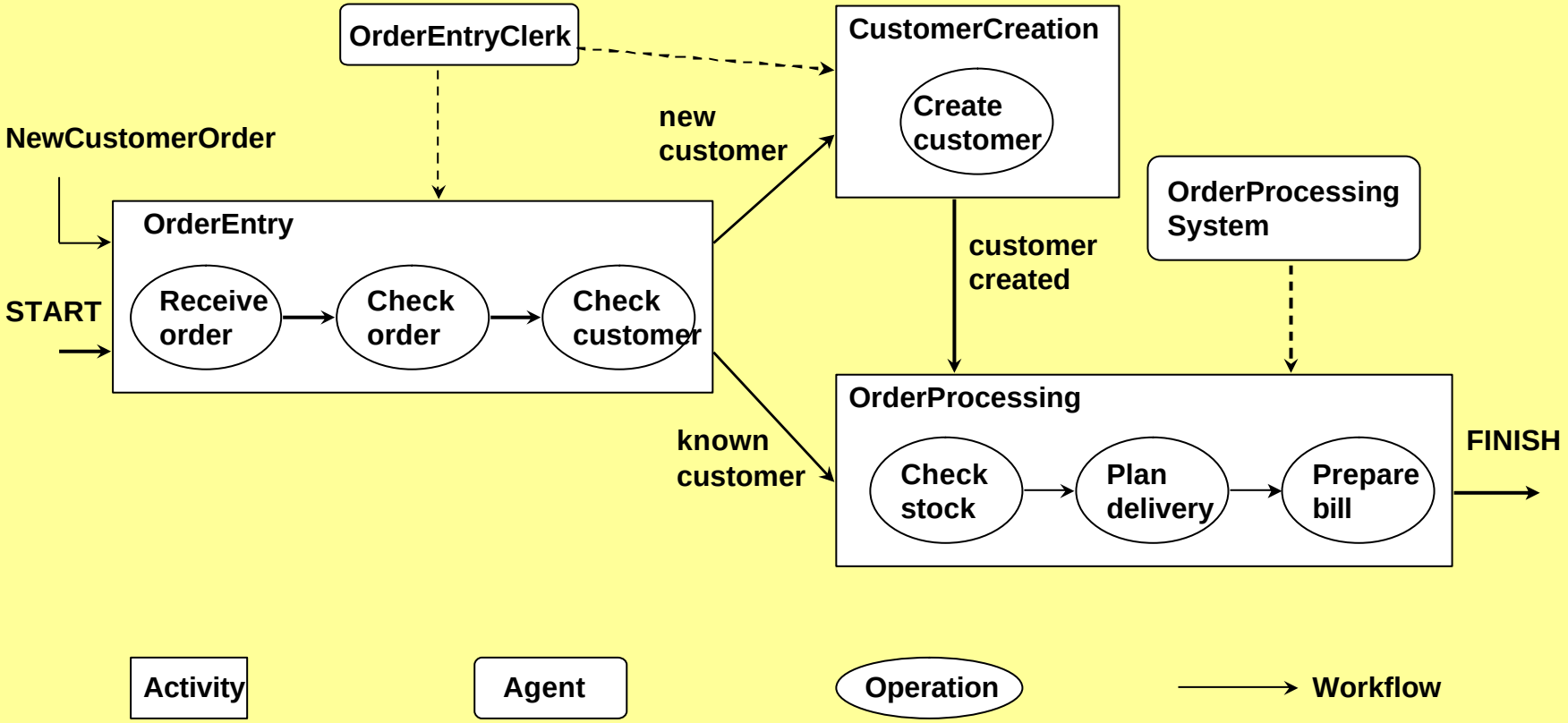
Enterprise/Divisions/Directions/Departments/Units/Positions

- **Organisation Unit:** lowest level decision centre or work position (i.e. role) assigned with tasks, responsibilities and authorities (on objects, agents and processes/activities)

- **Organisation Cell:** aggregation of organisation units and/or organisation cells into higher level decision centre with a manager, responsibilities and authorities

Example

Customer order processing (one process with three activities):



Example (cont'd)

FUNCTIONAL ENTITY OrderEntryClerk
- Type: Human
- InRealLife: M.S. SMITH
- ObjectView: OV-31 (* resource description*)
- Location: Bldg#1-Office#121
- Operations: ReceiveOrder (IN order);
  - CheckOrder (IN order, OUT: customer);
  - CheckCustomer (IN customer, OUT status);
  - CreateCustomer (IN customer, OUT Custid);
  - NotifyCustomer (OUT message);
  - ArchiveOrder (IN order)
  - UpdateCustFile (IN Custid, customer)

FUNCTIONAL ENTITY OrderProcessingSystem
- Type: Application
- ObjectView: OV-45
- Location: ABC/Grieg/OP-appli (UNIX machine)
- Operations: CheckStock (IN Sock#, ref#, qty,OUT status);
  - PlanDelivery (IN order, OUT deliverydate);
  - CreateMfgOrder (IN order; OUT MfgOrder);
  - PrepareBill (IN order, OUT customerbill);
  - SentBill (IN customerbill, OUT customerbill)

Example (cont'd)

OBJECT VIEW Order
RelatedObject: Order
Nature: Information
Properties:
    CustomerName: STRING [25].
    CustomerAddress: Address;
    OrderDate: Date;
    ItemLists: List [1:25] of ItemList;

EVENT NewCustomerOrder
Source: External
Timestamp: Arrival (Order)
ObjectView: OV-14 / Order

Example (cont'd)

PROCESS CustomerOrderProcessing
TriggeringEvents: NewCustomerOrder
ProcessBehaviour:
    WHEN (START WITH NewCustomerOrder) DO OderEntry
    WHEN (ES (OrderEntry) = new-customer) DO CustomerCreation
    WHEN (ES (CustomerCreation) = customer-created) DO OrderProcessing
    WHEN (ES (OrderEntry) = known-customer) DO OrderProcessing
    WHEN (ES (OrderProcessing) = any) DO FINISH

ACTIVITY OrderEntry
    FunctionInput: OrderFile
    ControlInput: Order
    ResourceInput: OrderEntryClerk
    FunctionOutput: OrderFile
    ControlOutput: {new-customer, known-customer}
    MeanDuration: 10 mn
    ActivityBehaviour:
        {OrderEntryClerk.ReceiveOrder (order);
        OrderEntryClerk.CheckOrder (order, customer);
        OrderEntryClerk.CheckCustomer (customer, status)}

Part III

Enterprise Modelling Ontologies

Ontology in IT

- **Ontology** is a branch of philosophy about the essence of things (i.e. what can exist in the world)

- In IT, *ontology* concerns the specification of a shared conceptualisation in a certain domain (Gruber, 92)
- An ***ontology*** is a formal description of entities and their properties, relationships, constraints, behaviours

Nota Bene: Ontology ? Taxonomy ? Thesaurus

Competency of an ontology = questions that the ontology is able to answer (Fox & Gruninger, 94)

Ontology: A trivial example

Ontology of circles (in geometry):

Definition:

A circle is defined by its center and radius in a 2D space

Formalisation:

Point (X, Y), $(X, Y) \in R^2$ (predicate)

Circle $(C1, r)$ $\in$ $C1$: Point $\in$ $r \in R$ (properties)

s.t. $r > 0$ (axiom)

Ontology example (cont'd)

However, a circle can be fully defined as well by:

- Three points *P1*, *P2* and *P3* taken on its circumference such that *P1* ? *P2* ? *P3*

- The set of points *P (X, Y)* that verify its canonical equation: $aX^2 + bY^2 = c$ (a, b, c being constants)

Conclusion: an ontology is not necessarly unique for a given universe of discourse.

Benefits of ontologies

- Expressiveness
- Formal precision (precise syntax; semantic axioms)
- Capitalises on extensive conceptual modelling experience (e.g. EER, Sowa's conceptual graphs)
- Increased cross-application generality and reusability
- Support for automated reasoning (rules, logical inference)
- Computationally constraining possible interpretations of data (intended model)

Ontologies

- Different types / generality levels of ontologies:
  - Foundational (upper) ontology (space-time, part-of, ...)
  - Domain and task ontologies
- Ontology specs express “micro-theories”
  - Lightweight: top-level structure often in formal diagrammatic form (e.g. UML-style or semantic networks)
  - Heavyweight: specified as finite axiom sets in (some subset of) first-order logic (FOL)
- Industry practice: often starts with metadata concept taxonomies (e.g. STEP, XML e-commerce standards)
  - but ontology encompasses much more

(Akkermans, 2001)

Ontology of time

- Uncertainty on points in time:

  time_point (t, min, max) ? *min* ? *t* ? *max*
- Partial order relation '<' (respect. '?')
- *SP(t)*: start point of interval *t*; *EP(t)*: end point of *t*
- EQ1: *(? t,t',p_1,p_2,p'_1,p'_2) strictly_before (t, t')* ?
  time_point(EP(t),p_1,p_2) ? *time_point(SP(t'),p'_1,p'_2) / p_2 < p'_1*
- EQ2: *(? t,t',p_1,p_2,p'_1,p'_2,p''_1,p''_2,p'''_1,p'''_2) during (t,t')* ?
  time_point(SP(t),p_1,p_2) ? *time_point(EP(t),p'_1,p'_2)* ?
  time_point(SP(t'),p''_1,p''_2) ? *time_point(EP(t'),p'''_1,p'''_2)*
  / p''_1 ? *p_1* ? *p'_2* ? *p'''_2*

(Gruninger & Fox, 94)

Enterprise Ontologies

- Why ontologies are important for EM?

1. Need for a formal *enterprise modelling language*
2. Need for agreed *theoretical foundations*
3. Need for *knowledge sharing* among intelligent agents / modelling tools

- Ontology projects in enterprise modelling:
  - TOVE project at Univ. of Toronto (Fox & Gruninger)
  - The Enterprise Ontology, Univ. of Edinburgh (Ushold)
  - IDEF 5 (KBSI & State Univ. of Texas at Austin)
  - I* methodology, Univ. of Toronto (Yu & Mylopoulos)
  - PSL, NIST, USA

CIMOSA Ontology principles

Ontology description method: Two parts:

- *Terminology definition:* semantic networks or conceptual graphs
- *Concept specification:* Order-sorted algebra

Sort: Defined by:

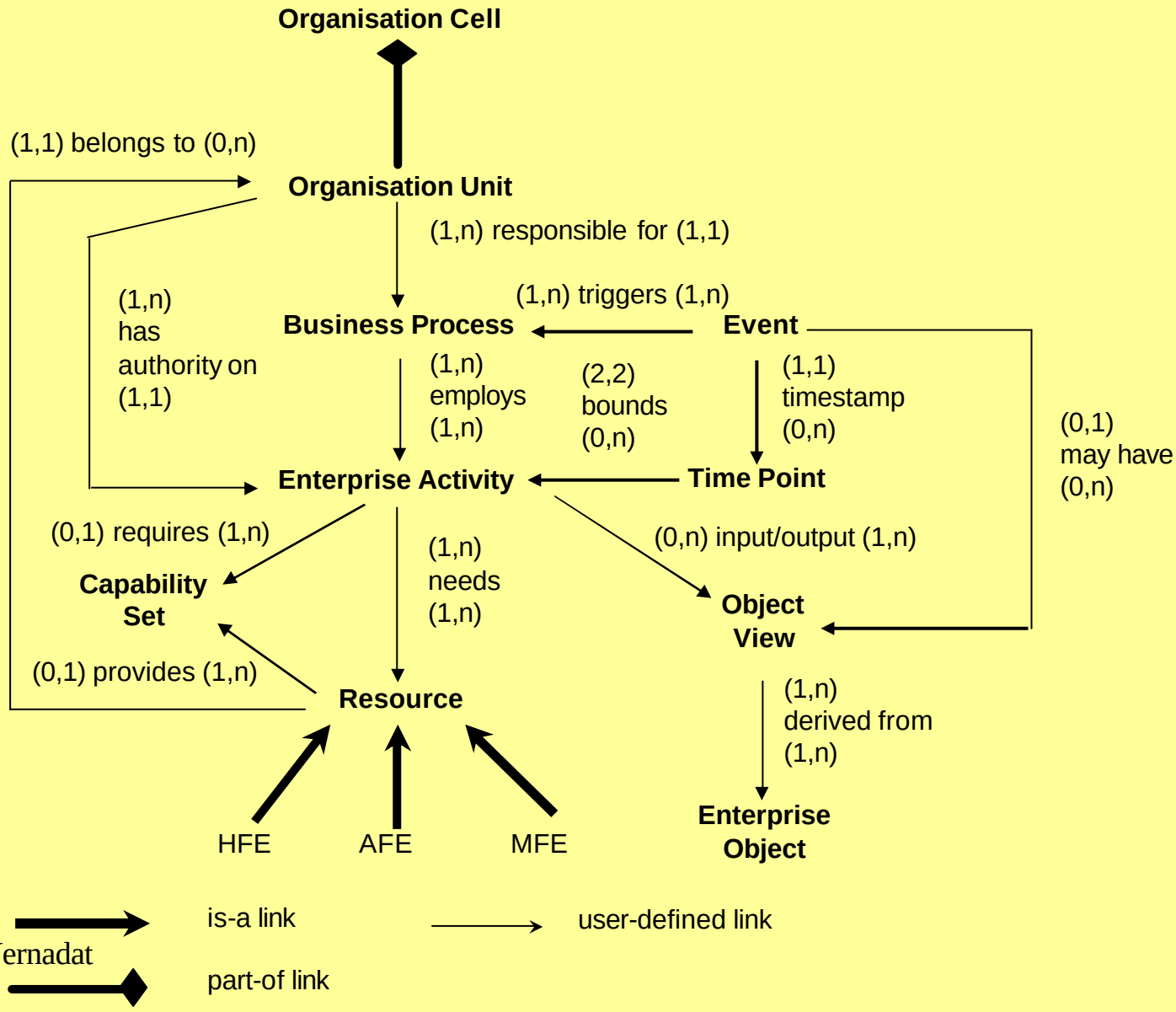
- the sort *name*
- a set of typed *variables*
- a set of *operations* on variables
- a set of *axioms* defining the semantic rules
- a set of *relationships* defining semantic or relational links of this concept with other concepts of the ontology

CIMOSA Ontology (cont'd)

Three kinds of links can be used in sorts to relate ontological concepts to one another:

- *semantic relationships*
  - isa link (property inheritance mechanism)
  - part-of link
- *user-defined relationships*
  - named relationships (n-ary)
- *terminology relationships* e.g. 'is-equivalent-to', 'is-related-to', 'may-be-related-to', 'is-synonym-to', etc.

CIMOSA Ontology: terminology definition



CIMOSA Ontology: Concept specification

Event: Events are facts (solicited or not) indicating a change in the system state.

SORT Event

IMPORT Object_View, Enterprise_Activity, Resource, REAL, BOOLEAN

VARIABLES

Source: Resource ? Enterprise_Activity ? {external}

ObjectView: Object_View

TimeStamp: REAL

OPERATIONS

CreateEvent: Event ? BOOLEAN

Active: Event ? BOOLEAN

AXIOMS

(? *e* ? Event) *CreateEvent* (*e*) ? TimeStamp (e) = *SP(e)*

(? *e* ? Event) *Active* (*e*) ? *defined* (TimeStamp(*e*))

CIMOSA Ontology: Concept specification

Enterprise activity: Enterprise activities are the locus of action, i.e. they transform an input state into an output state using resources and time within the course of a process.

SORT Enterprise_Activity

IMPORT Event, Object_View, Resource, LABEL, REAL

VARIABLES

Function_Input: *P*(Object_View)

Control_Input: *P*(Object_View)

Resource_Input: *P*(Resource)

Function_Output: *P*(Object_View)

Control_Output: LABEL ? *P*(Event)

Resource_Output: Object_View

Ending_Statuses: *P*(LABEL)

CIMOSA Ontology: Concept specification

Enterprise activity (cont'd):

Minimum_Duration: REAL

Maximum_Duration: REAL

Required_Capabilities: Capability_Set

Activity_Behaviour: An_Algorithm

Used_By: *P*(Business_Process)

OPERATIONS

Start: Enterprise_Activity ? BOOLEAN

Finish: Enterprise_Activity ? BOOLEAN

Duration: Enterprise_Activity ? REAL

Ending_Status: Enterprise_Activity ? LABEL

P(*S*) = power set of *S* = 2^S

CIMOSA Ontology: Concept specification

Enterprise activity (cont'd):

AXIOMS

(? *a* ? Enterprise_Activity) Function_Input (*a*) ?
Control_Input (*a*) ? Function_Output (*a*) ? ? ?
Function_Input (*a*) ? Control_Input (*a*) ? ? ?
Resource_Input (*a*) ? ? ? Control_Output (*a*) ? ? ?
Minimum_Duration (*a*) ? Maximum_Duration (*a*)

(? *a* ? Enterprise_Activity) *Start* (*a*) ?
defined (preconditions (Activity_Behaviour (a)))

(? *a* ? Enterprise_Activity) *Finish* (*a*) ?
defined (Ending_Status (*a*)) ?
Ending_Status (*a*) ? Ending_Statuses (*a*) ?
Ending_Status (*a*) ? Control_Output (*a*)

(? a ? Enterprise_Activity) *Duration (a) = EP(a) - SP(a)*

TOVE Ontology principles

- To support reasoning in industrial environments

  Basis for NIST's PSL (Process Specification Language)

- The goal of the TOVE (TOronto Virtual Enterprise) Enterprise Modelling project is to create enterprise models that not only answer queries with what is explicitly represented, but also be able to deduce answers to queries.
  - Activity ontology
  - Resource ontology
  - Cost ontology
  - Quality ontology

TOVE Activity Ontology

- Objectives:
  - Temporal projection – resources and activities
  - Planning and scheduling
  - Execution monitoring and external events
  - Time-based competition
- Specification formalism:
  - First-Order Logic (implemented in Prolog)
- Foundations
  - Situation calculus (Reiter 91)

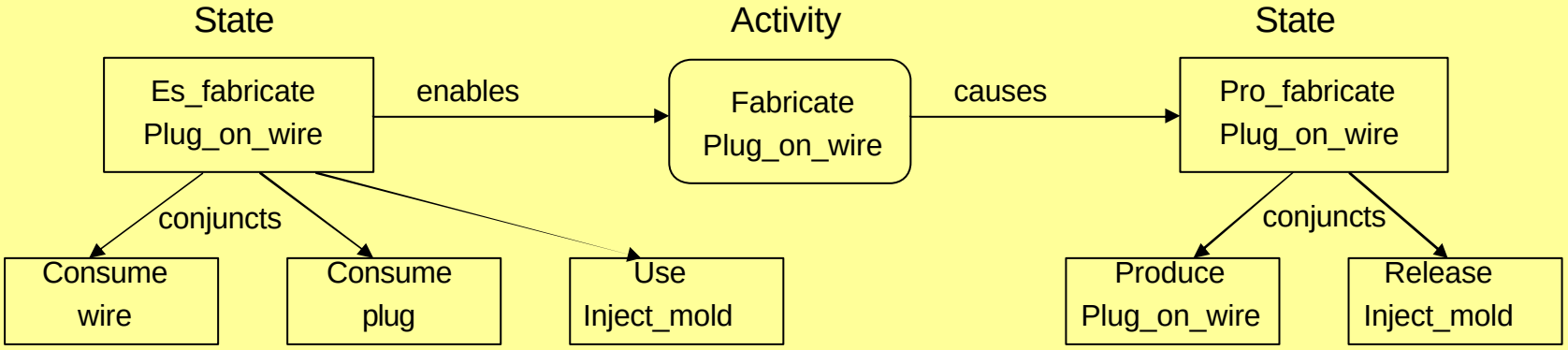
TOVE Activity Ontology

- **Situation calculus:** is a sorted second order language with equality

  Five domain sorts $<A, S, F, T, D>$ respect. for
  - action types
  - situations
  - fluents (or control flows)
  - time, and
  - arbitrary objects

TOVE Activity Ontology

- **Situation calculus:** provides a semantics to an ontology of activity and state
  - There is an initial situation s_0
  - The system evolves from one situation *s* to another *s'* when an action *a* is performed (stochastic automata)
  - The structure of situations is that of a tree
    (root = s_0; each branch = one possible future)
  - Activity-State model (sub-states allowed)



TOVE Activity Ontology

- **Situation calculus:** predicates and axioms
  - **Predicates:**
    - *s* ? *s'*: denotes that situation *s* precedes situation *s'*
    - *Poss* (*a, s*): true if action *a* can be performed in situation *s*
    - *do* (*a, s*): returns the name of situation that results from performing action *a* in situation *s*
    - *do* (*a, s, s'*): denotes that if action *a* is done in situation *s*, then *s'* is one of the possible situations reached (complex activities)
    - *actual* (*s*): true if s is the actual situation
    - *occurs* (*a, s*): true when action *a* is performed from *s*

TOVE Activity Ontology

- **Situation calculus:** predicates and axioms
  - **Axioms:**

S_0 has no antecedent: $(\forall a, s)\ S_0 \neq do\ (a, s)$

Every situation has a unique predecessor:

$$(\forall a_1, a_2, s_1, s_2)\ do\ (a_1, s_1) = do\ (a_2, s_2) \supset a_1 = a_2$$

Second-order induction axiom for situations:

$$(\forall P)\ (P(S_0) \supset (\forall a, s)\ (P(s) \supset P(do\ (a, s)))) \supset (\forall s)\ P(s)$$

The initial situation precedes all other situations:

$$(\forall s) \supset s < S_0$$

The successor of a situation is later than the situation:

$$(\forall a, s_1, s_2)\ s_1 < do\ (a, s_2) \supset (Poss\ (a, s_2) \supset s_1 \sqsubset s_2)$$

TOVE Activity Ontology

- **Situation calculus:** predicates and axioms
  - **Notion of causality:** what holds after performing an action

  Successor state axiom: derives successor state for each fluent

  $(\forall a, s)\ Poss(a, s) \rightarrow [holds(R, do(a, s)) \leftrightarrow \gamma^{+}_{R}(a, s) \vee (holds(R, s) \wedge \neg \gamma^{-}_{R}(a, s))]$

  $\gamma^{+}_{R}(a, s)$ (respect. $\gamma^{-}_{R}(a, s)$) is a simple formula specifying the conditions under which an action *a* asserts (respect. falsifies) the fluent *R*.

  Occurrence of actions:

  $actual(S_0)$

  $(\forall a, s)\ actual(do(a, s)) \rightarrow actual(S) \wedge Poss(a, s)$

  $(\forall a_1, a_2, s)\ actual(do(a_1, s)) \wedge actual(do(a_2, s)) \rightarrow a_1 = a_2$

  $(\forall a, s)\ occurs(a, s) \rightarrow actual(do(a, s))$

  (actions performed along the actual line)

PSL: Process Specification Language

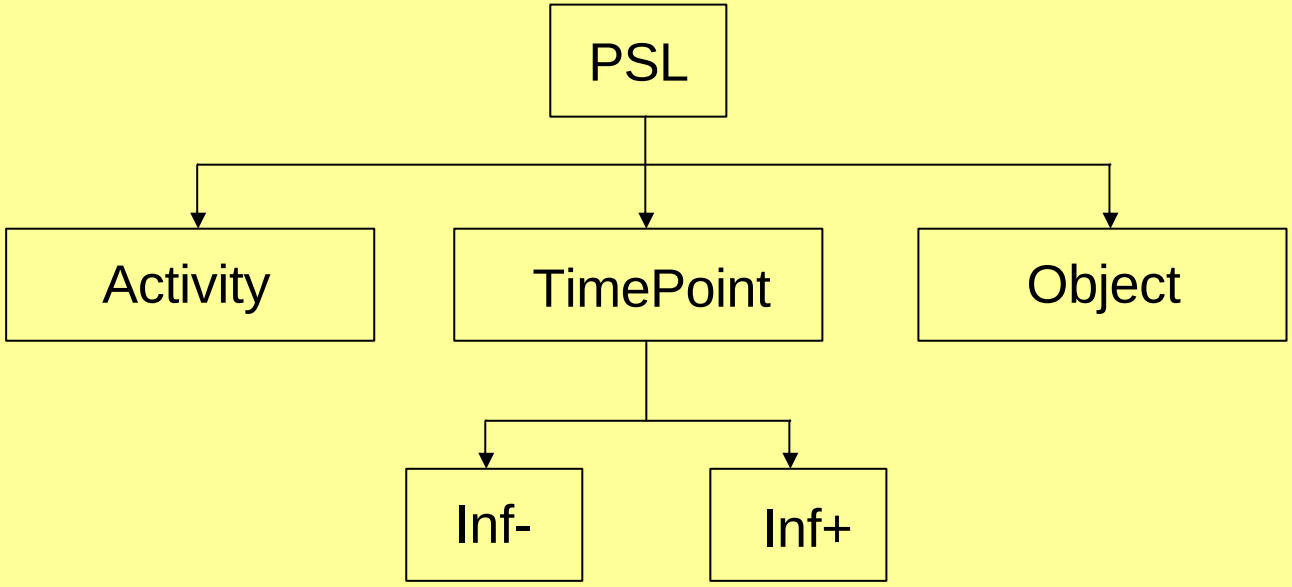
- Developed under sponsorship of NIST
- International collaboration
- Industrial Automation standardisation activity:

  ISO TC 184/SC4

- Aims:
  - formal and neutral representation of manufacturing processes
  - Interlingua to exchange process information among industrial applications

What is a process in PSL?

- A **process** is one or more activities that occurs over a period of time in which objects participate.



PSL Core

- **Intuition 1:**

  There are four kinds of entities required for reasoning about processes -- activities, activity occurrences, timepoints, and objects.
- **Intuition 2:**

  Activities may have multiple occurrences, or there may exist activities that do not occur at all.
- **Intuition 3:**

  Timepoints are linearly ordered, forwards into the future, and backwards into the past.
- **Intuition 4:**

  Activity occurrences and objects are associated with unique timepoints that mark the begin and end of the occurrence or object.

PSL Core

- **Entities**
  - activity
    activity-occurrence,
    timepoint,
    object
- **Relations**
  - before, between,
    beforeEq, betweenEq,
    is-occuring-at, participates-in, exists-at
- **Functions**
  - beginof, endof

Examples of PSL Axioms

Axiom 1 *The before relation only holds between timepoints.*

(forall (?t1 ?t2) (implies (before ?t1 ?t2) (and (timepoint ?t1) (timepoint ?t2))))

Axiom 2 *The before relation is a total ordering.*

(forall (?t1 ?t2) (implies (and (timepoint ?t1) (timepoint ?t2)) (or (= ?t1 ?t2) (before ?t1 ?t2) (before ?t2 ?t1))))

Axiom 3 *The before relation is irreflexive.*

(forall (?t1) (not (before ?t1 ?t1)))

Axiom 4 *The before relation is transitive.*

(forall (?t1 ?t2 ?t3) (implies (and (before ?t1 ?t2) (before ?t2 ?t3)) (before ?t1 ?t3)))

Axiom 9 *Everything is either an activity, activity occurrence, timepoint, or object.*

(forall (?x) (or (activity ?x) (activity_occurrence ?x) (timepoint ?x) (object ?x)))

The Enterprise Ontology

- Developed at AI Applications Institute (AIAI), Univ. of Edinburgh, 1995-1998
- Together with IBM
- See http://www.aiai.ed.ac.uk/project/enterprise/
- Collection of terms and definitions relevant to business enterprises
- The major role of the Enterprise Ontology is to act as a communication medium, in particular, between:
  - Different people (users and developers) across diff. enterp.
  - People and implemented computational systems
  - Different implemented computational systems (e.g. ERP, DBMS, spreadsheets…)

The Enterprise Ontology

Ontology organisation:

1. ***Meta-Ontology and time*** – terms used to define the terms of the Ontology (e.g. Entity, Relationship, Role) and terms related to time (e.g. Time-Interval)
2. ***Activity, Plan, Capability and Resource*** – terms related to processes and planning (e.g. Activity, Planning, Authority, Resource, Allocation)
3. ***Organisation*** – terms related to how organisations are structured (e.g. Person, Legal Entity, Organisation Unit)
4. ***Strategy*** – terms related to high level planning for an enterprise (e.g. Purpose, Mission, Decision, Critical Success Factors)
5. ***Marketing*** – terms related to marketing and selling goods and services (e.g. Sale, Customer, Price, Brand, Promotion)

The Enterprise Ontology: overview

ACTIVITY etc.	ORGANISATION	STRATEGY	MARKETING	TIME
Activity	Person	Purpose	Sale	Time Line
Activity Specification	Machine	Hold Purpose	Potential Sale	Time Interval
Execute	Corporation	Intended Purpose	For Sale	Time Point
Executed Activity Specification	Partnership	Purpose-Holder	Sale Offer	
T-Begin	Partner	Strategic Purpose	Vendor	
T-End	Legal Entity	Objective	Actual Customer	
Pre-Condition	Organisational Unit	Vision	Potential Customer	
Effect	Manage	Mission	Customer	
Doer	Delegate	Goal	Reseller	
Sub-Activity	Management Link	Help Achieve	Product	
Authority	Legal Ownership	Strategy	Asking Price	
Activity Owner	Non-Legal Ownership	Strategic Planning	Sale Price	

The Enterprise Ontology: overview

ACTIVITY etc.	ORGANISATION	STRATEGY	MARKETING	TIME
Event	Ownership	Strategic Action	Market	
Plan	Owner	Decision	Segmentation Variable	
Sub-Plan	Asset	Assumption	Market Segment	
Planning	Stakeholder	Critical Assumption	Market Research	
Process Specification	Employment Contract	Non-Critical Assumption	Brand	
Capability	Share	Influence Factor	Image	
Skill	Shareholder	Critical Influence Factor	Feature	
Resource		Non-Critical Influence Factor	Need	
Resource Allocation		Critical Success Factor	Market Need	
Resource Substitute		Risk	Promotion	
			Competitor	

The Enterprise Ontology: overview

1. **Informal EO:** the natural language version

Enterprise Activity Example:

"The concept of ACTIVITY is closely linked with the idea of the DOER, which EXECUTEs an ACTIVITY SPECIFICATION by performing the specified ACTIVITIES. A DOER may be a PERSON, ORGANISATIONAL UNIT or MACHINE."

"The ability of a POTENTIAL ACTOR to be the DOER of an ACTIVITY is denoted by CAPABILITY (or SKILL if the DOER is a PERSON). ACTORS may have other Roles in respect of an ACTIVITY such as ACTIVITY OWNER."

The Enterprise Ontology: overview

1. **Informal EO:** the natural language version

ACTIVITY: something done over a particular TIME INTERVAL. The following may pertain to an ACTIVITY:

- **Has PRE-CONDITIONS**
- **Has EFFECT(S)**
- **Is performed by one or more DOERS**
- **Is decomposed into more detailed SUB-ACTIVITIES**
- **Entails use and/or consumption of RESOURCES**
- **Has AUTHORITY requirements**
- **Is associated with and [ACTIVITY] OWNER**
- **Has a measured efficiency**

The Enterprise Ontology: overview

2. Formal EO:

- Use the Ontolingua language (based on KIF) from Stanford University, CA

```
(Define-Frame Activity
    :Own-Slots ((Documentation "Something done over a particular Time-Range. The following
    may pertain to an Activity:
    * is performed by one or more Actual-Doer s;
    * is decomposed into more detailed Sub-Activity s;
    * Can-Use-Resource s; * An Actor may Hold-Authority to perform it;
    * there may be an Activity-Owner;
    * has a measured efficiency. ")
    (Instance-Of Class) (Subclass-Of Activity-Or-Spec))
    :Template-Slots
    ((Actual-Activity-Interval (Minimum-Cardinality 0) (Cardinality 1)
        (Value-Type Time-Range))
    (Actual-Pre-Condition (Minimum-Cardinality 1) (Value-Type Pre-Condition))
    (Actual-Effect (Minimum-Cardinality 1) (Value-Type Effect))
    (Activity-Status (Minimum-Cardinality 1) (Value-Type Activity-State))))
```

References on EM ontologies

- Gruber, T. (1992) Ontolingua: A Mechanism to Support Portable Ontologies. Res. Rep. KSL-91-66, Stanford Knowledge Systems Lab., Final Version, Stanford University, Stanford, CA.

  http://www-ksl.stanford.edu
- Grüninger, M. and M.S. Fox (1994) An activity ontology for enterprise modelling. Third IEEE Workshop on Enabling Technologies: Infrastructures for Collaborative Enterprises (WET ICE'94), Morgantown, West Virginia. http://www.ie.utoronto.ca/EIL/
- Fox, M.S. and M. Grüninger (1994) Ontologies for enterprise integration. Second Int. Conf. on Cooperative Information Systems, Toronto, May. pp. 82-89.
- Benjamin, P.C., C.P. Menzel, R.J. Mayer and N. Padmanaban (1995) Toward a method for acquiring CIM ontologies. *Int. J. of Computer-Integrated Manufacturing,* **8**(3): 225-234.
- Uschold, M., M. King, S. Moralee and Y. Zorgios (1998) The Enterprise Ontology. Res., Rep. Artificial Intelligence Applications Institute, Edingburgh, Scotland, June. http://www.ed.ac.uk/

CONCLUSION

- Enterprise Modelling constructs
  - Significant work already achieved – tools available
  - Existence of norms/standards for well-structured processes
  - Constructs needed for ill-structured processes
  - “Soft issues” still poorly addresses (e.g. BP rationale, skills/competencies, mission, goals, strategic objectives…)
- Enterprise Modelling ontology
  - Still a research area
  - Which formalism is best?
  - Can we agree on a common EM ontology?
  - Or do we have to map different tool/domain ontologies?