IKERLAN

TECHNOLOGICAL RESEARCH CENTRE

A CSP based Distributed Product Configuration System

Fernando Eizaguirre and **Martín Zangitu** and
Josune de Sosa and **Karmele Intxausti**

Pº J.M. Arizmendiarrieta, 2
20500 Arrasate — Mondragón
(Gipuzkoa - Spain)

Tel.: 943 71 24 00
Fax: 943 79 69 44

IKERLAN Technological Research Center

Profile & Mission

- **Private not-for-profit institution.**

- **Created in 1974, in the heart of MCC Group**

  (Co-operative enterprises + MGEP + CL)

- **Member of INNOBASQUE and FEDIT.**

- **Member of the IK4 Alliance** .

- **250 workers**

To help reinforce companies' innovation capacity to improve their competitiveness, thereby helping to boost the country's economic and social development, through the generation and application of technology and knowledge.

Headquarters

Arrasate

Miñano

Technology generation and transfer

Assimilation and generation

- **Research projects:**

  Basque Government: ETORTEK, SAIOTEK

  DFG: Gipuzkoa Provincial Council

  AGE: Research projects, PROFIT, CENIT

  UE: 7th Framework Programme, EUREKA

Transfer and dissemination to companies

- **Contracted R&D projects**
- **Technological consulting**
- **Personnel training and transfers**

Business Units

New product development

- Calculation, simulation and conceptual design of mechanisms.
- Acoustic noise.
- Electronic management and control systems.
- Sensors.
- Connectivity and telecommunications networks.
- Dynamic and reconfigurable software systems.
- Microtechnologies for industry and life sciences.

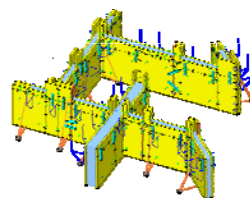## Generation and rational use of energy

- Combustion, burners and their applications.
- Solid oxide fuel cells.
- Climate control and refrigeration.
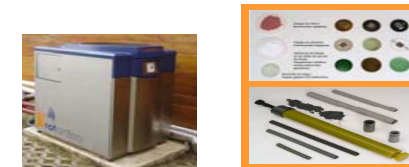
Industrialisation and exploitation of the product life cycle

- Organisation of the innovation.
- Redesign of enterprise processes in a sustainable company.
- Innovative software for industrial processes

A CSP based Distributed Product Configuration System

Experience in Product Configuration

- **Long Experience in Centralised Configuration Systems for Manufacturing Companies**
  - **Buses**
  - **Elevators**
  - **Domestic Appliances**
  - **Construction**

- **Current Industrial Version of the Configuration System**
  - **No CSP technology incorporated**
  - **Centralized**
  - **No Web interfaces**

- **Research Project to incorporate**
  - **CSP technology**
  - **Distributed Product Model**
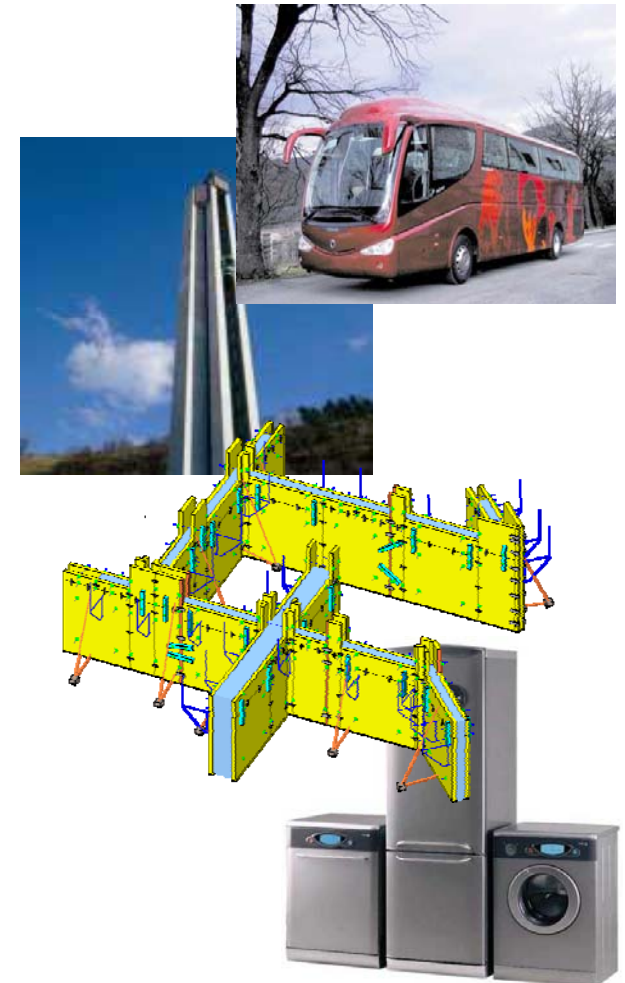
Targeted scenario

- **Configurator Interface accessible via Web with light clients**
  - Clients access the configurator via web
  - Typical example of configuration can be found in domestic appliances sector

- **Small – medium size problems accessible via Web**
  - Configuration problems are small – medium
  - A washing machine has "tenths" of variables
  - An elevator can have up to one hundred of variables but no more

- **Product Model Distributed across different nodes**
  - Components of the product supplied by different companies

Main Functional Requirements to be implemented

- **Consistent Selection of Values**
  - At each configuration point only consistent values (part of a valid solution) can be selected
  - After each selection, inconsistent values must be labeled as no longer consistent

- **Selection of Values currently labelled as no longer consistent**
  - At a given point in the configuration process the user wants to select an inconsistent value
  - The system should propose previous decisions to be changed

- **Minimum and Maximum Cost in Real-time**
  - As the configuration process goes-on the system must show minimum and maximun cost
  - Real-time is desirable (few seconds)

- **Automatic Termination of the solution**
  - The user ask to the system to complete the solution

Distributed Product Model
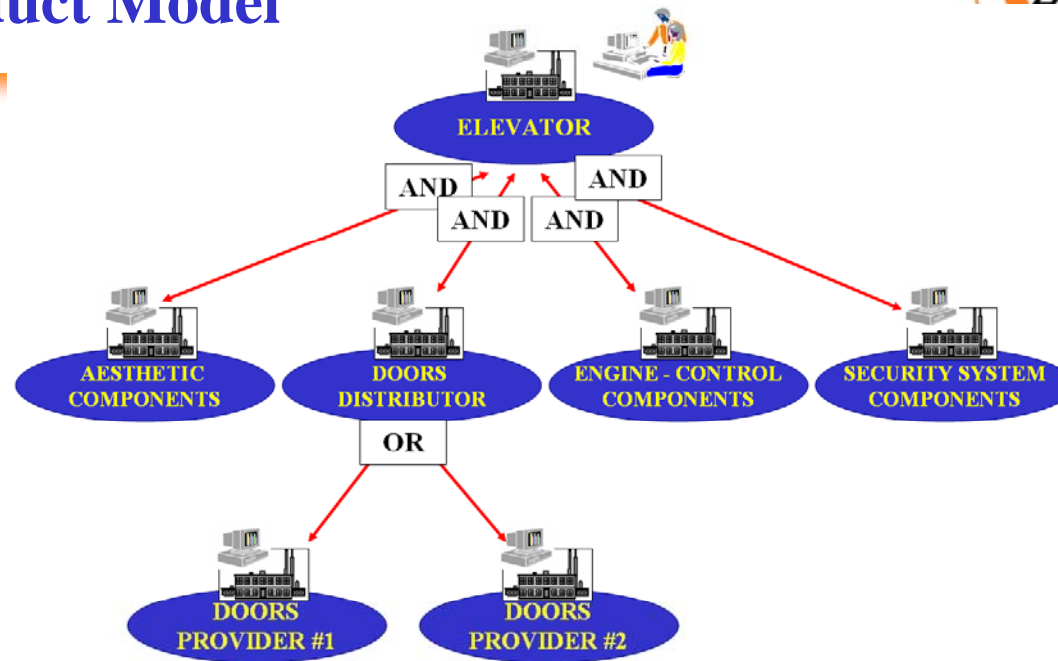
- **Product Model Distributed in a Hierarchy of Nodes / Sub Nodes**
  - Sub Nodes typically represents suppliers of components
  - Each Sub Node may represent a different company
  - Distributors may be represented as OR nodes

- **Private Constraint Network**
  - Interface of each node defined in terms of Variables
  - Constraints are private to each company (private know-how)

-> Typical complexity of each node

Distributed Product Model as DisCSP

PUBLIC **VARIABLES**

A, B, I, J, N, P

I, J, K   M, N, O, P

NODE

VARIABLES IMPORTED
From
SUB NODE #1

VARIABLES IMPORTED
From
SUB NODE #2

PRIVATE
CONSTRAINT
NETWORK

Remote Equality
Constraint

Remote Equality
Constraint

PUBLIC **VARIABLES**

PUBLIC **VARIABLES**

I, J, K

M, N, O, P

SUB NODE

SUB NODE

PRIVATE
CONSTRAINT
NETWORK

PRIVATE
CONSTRAINT
NETWORK
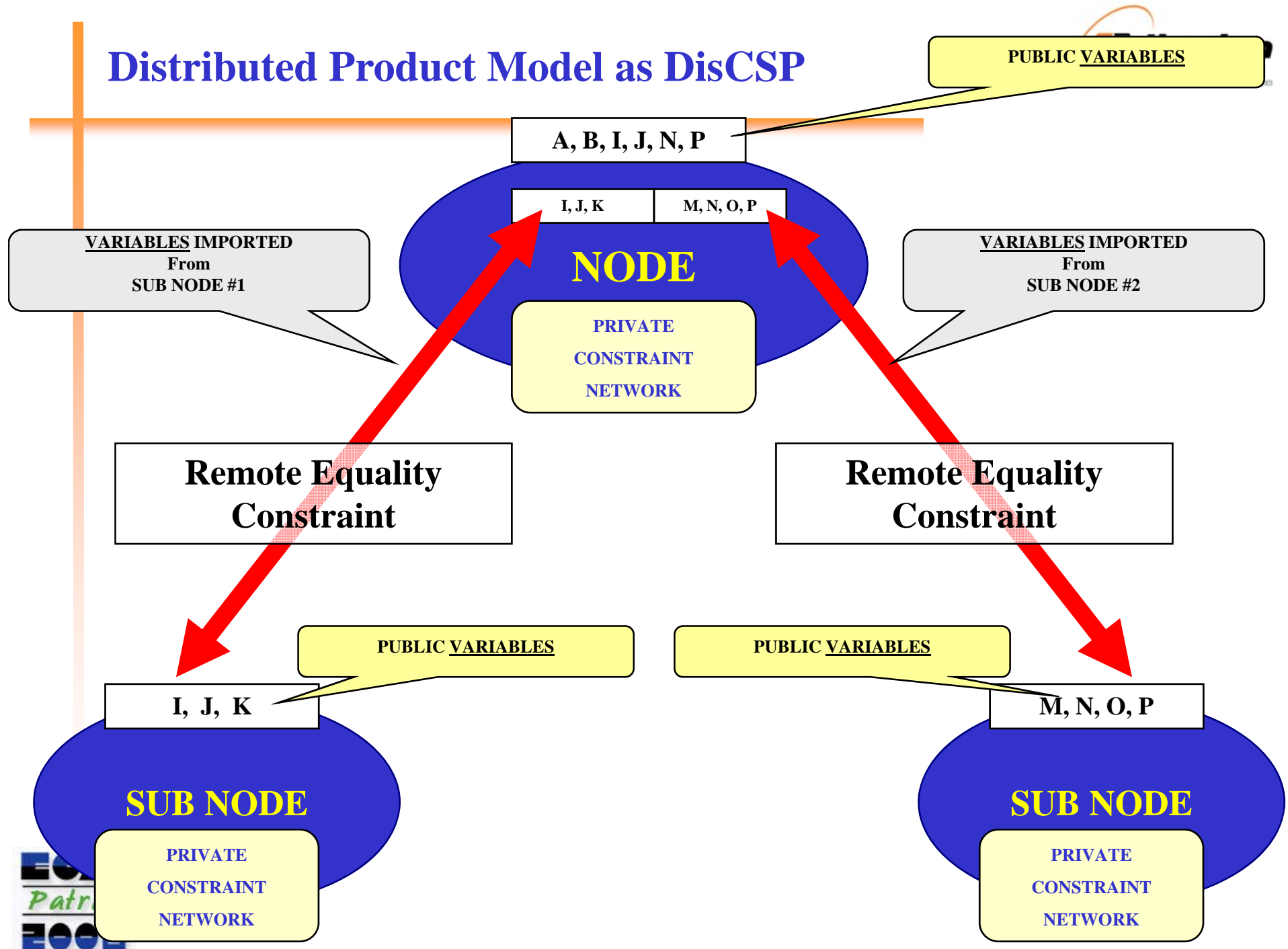
State of the Art in Distributed CSP

- **Synchronous Vs. Asynchronous**
    - **Revised some Asynchronous Algorithms**
    - **Not easy to implement in a Client-Server environment as the one described here**
    - **In the presence of message delays efficiency may be not better than Synchronous Alg.**

- **CAWICOMS**
    - **Synchronous backtracking implementing an "extended forward checking" via WService**
    - **We followed a similar approach but using sockets for efficiency (security concerns)**

- **Concurrent Search with Synchronous Backtracking over Non-Intersection Sections**
    - **Algorithm proposed by R. Zivan and A. Meisels**
    - **Very interesting algorithm to be incorporated in a next version**
    - **Efficiency of concurrency but with the simplicity of Synchronous Algorithms**

- **Constraint Compilation (for real-time)**
    - **Very interesting for real-time**
    - **Not well adapted for distributed problems or environments with changing constraints**

Remote Equality Constraint: extended FC

ikerlan
ik4 research alliance

Calling constraint engine receives new domains or backtracking message and keeps working
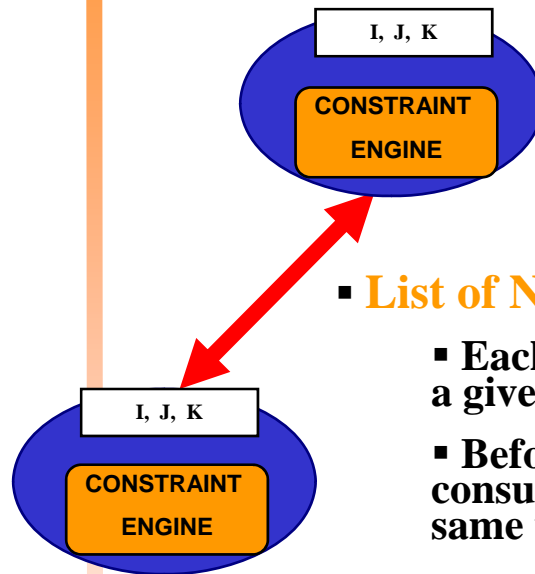
Whenever the domain of I, J, K, is modified, the constraint is invoked

I, J, K

CONSTRAINT ENGINE

The constraint engine collects the current domains of I, J, K and sends current domains to subnode's constraint engine for checking consistency

SOCKET Connect

Calling constraint engine waits Synchronously for answer

Remote Equality Constraint

When the subnode's constraint engine receives the domains, checks with its local constraint network (could involve subnodes)

SOCKET Listening

CONSTRAINT ENGINE

The constraint engine answers back with the new (possibly pruned and arc-consistent) domains, or with a backtrack message if any domain is empty

I, J, K

The constraint engine waits for new requests

ECAI
Patras
2008

Remote Equality Constraints: Speeding Up

I, J, K

CONSTRAINT ENGINE

I, J, K

CONSTRAINT ENGINE

- **List of NoGoods and List of Goods**
  - Each remote equality constraint stores the list of all past requests/answers (in a given interaction of the user)
  - Before requesting something to a configurator of a subnode, the constraint consults its list to check if that request was done before, avoiding repeating the same work

- **Improvement of efficiency is very important**
  - Backtracking repeats the same work many times (trashing) but …
  - Subnodes are not called again and again …

- **Price to be paid**
  - Memory

Algorithm for selecting inconsistent values

- **Configuration Problem**
    - **Variables**
        - *VarA, VarB, VarC, VarD*
    - **Constraints**
        - *C1, C2, C3, C4*
- **Current State of Configuration Process (User's decisions or constraints)**
    - *VarA = 7, VarB = 8, VarC = 2*
- **The user wants to select**
    - *VarD = 12*, unfortunately this value is no longer consistent with previous decisions
- **To help the user the configurator tries to find a solution to the problem**
    - *VarD = 12*
    - *$KeepD_1$ => (VarA = 7)*   *(reified constraint)*
    - *$KeepD_2$ => (VarB = 8)*   *(reified constraint)*
    - *$KeepD_3$ => (VarC = 2)*   *(reified constraint)*
    - Find a solution maximising the number of variables *$KeepD_i$* that equals to True
- **The configurator proposes to the user to**
    - Maintain decisions for which *$KeepD_i$* variables is True
    - Eliminate decisions for which *$KeepD_i$* variables is False

Current State of Implementation

- **Distributed Configurator**
  - **Based in SICStus Prolog clpFD**
  - **Remote Equality Constraint Implements an "extended" forward checking**
  - **Allows clients to interact with the root node via web interface**

- **Distributed Model Editor**
  - **Allows the administrator defining the distributed nodes**
  - **Generates automatically the SICStus Prolog clpFD to be deployed in each node**
  - **Generates automatically a simple web-based interface of the root node**

Summary and Next Steps

Summary

- **This paper has presented a Disributed Product Configurator**
- **Is the current result of an ongoing research project**
- **The project is not aimed at developing new techniques or algorithms in CSP**
- **But consists of a practical application of a variety of known techniques**
- **The system is built on top of SICStus Prolog clpFD**

Next Steps

- **Implement a Concurrent Search Algorithm with Synchronous Backtracking**
  - **Extend this prototype with concurrent algorithm of R. Zivan and A. Meisels**
  - **Takes advantage of concurrency but keeping synchronicity**
- **Testing System Performance with problems of different size and complexity**
  - **Generate random problems of different size and complexity**
  - **Test exhaustively under different network conditions (LAN, VPN, Internet)**

IKERLAN

TECHNOLOGICAL RESEARCH CENTRE

**Pº J.M. Arizmendiarrieta, 2
20500 Arrasate – Mondragón
(Gipuzkoa)**

**Tel.: 943 71 24 00
Fax: 943 79 69 44**

**Fernando Eizaguirre
feizaguirre@ikerlan.es**