BETREUT VOM

Projektträger
Forschungszentrum
Karlsruhe (PTKA)

GEFÖRDERT VOM

Bundesministerium
für Bildung
und Forschung

XXXXX Machine configuration

Uwe Lesta
SBS-Softwaresysteme GmbH

Ulm

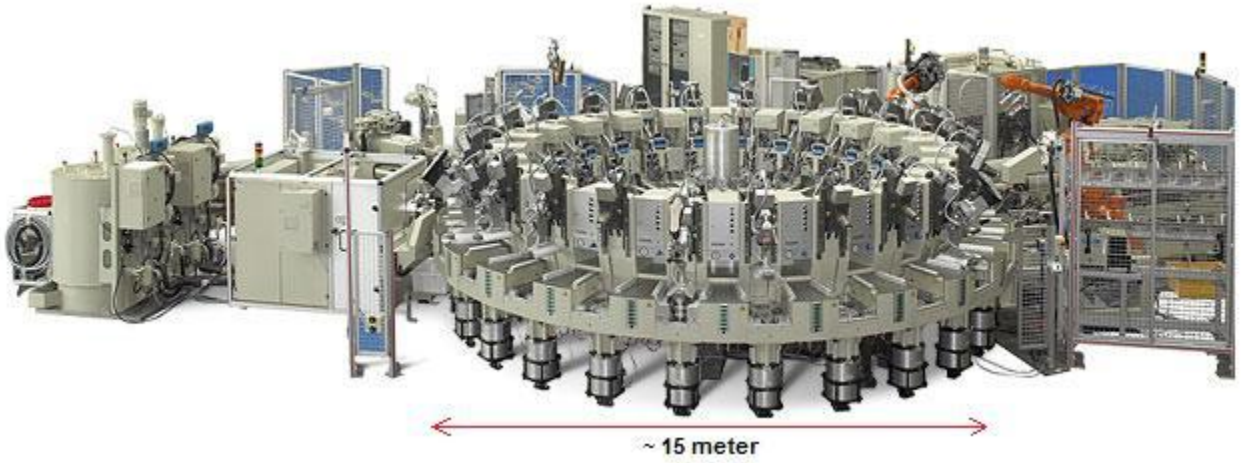
07. October 2009

Overview

- Introduction
- Requirements
- Machine description
- Solution approaches

Introduction

Machines are build individually for a customer. For example machines to build plastic parts.



The purchaser defines the machine by:

- The parts which should be produced including the materials (Rubber, PU, …).
- The place where the machine should work. (country, factory building, …)
- The cost per piece, pieces per hours.

Introduction

The sales divisions have to make an offer from these requirements.

- In short time
- In a state of uncertainty if it is possible.
- For a fixed prise for the machine and the cost per piece.

Engineers from the construction division are helping to master this task. They have to make 30 to 70 offers for one order.

A software tool should help.

The goal of the project is to have some pieces of software which can be used to configure technical products. Primary for an offer.

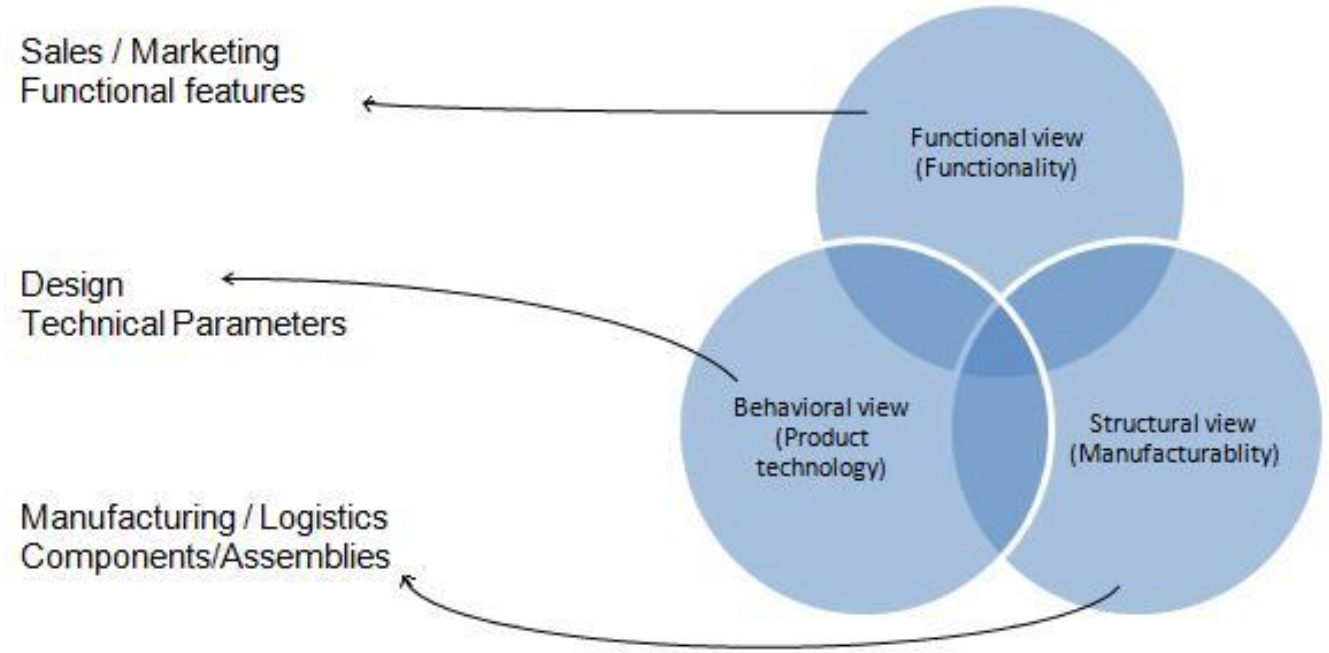
Requirements

The user of the software must be able to change the knowledge about the machine.

- Editable (facts, components, times, prices, …)
- Textual representation
- Modules
- Documentation / Explanation
- Different Views

Product family architecture

The software should support different views to the knowledge with a different vocabulary.
Because the software should create an offer the functional view is emphasised.



A good knowledge model

If the structures of the knowledge retrieve the structures of its representation and if the inference engine is based on the utilisation of these structures.

This ensures transparency.

On the other hand it might be useful to compile the knowledge representation into an other format for faster configuration.

Machine description

The base of the machine is a round table with N={24, 28, 32, 38, 44, 48, 60} places.

There are several base types with a different set of N which are able to handle a subset of plastics.

On each place on the round table is a form for a plastic (e.g. for Polyurethane PU). The forms can be changed e.g. for other parts or sizes.

To produce a piece there are m process steps PS necessary in a defined order.

Each process step has a process time PT and a time behind BT. These times depend on the material and the form. Each PS is executed on a defined position around the table.

Machine description

A process step can be executed by a
- Machine
- Human
- Machine or human

A machine can execute one process step.

Machine is_one_of [robot, extruder, …]

A robot can do one or two process steps. Each robot has a price, a time to execute a process step and more attributes. Some options can change the execution time. The sum of the parts is the price of the robot.

Machine description

A robot consists of (simplified):
base_type, control, security, options.

base_type is_one_of
e38534, 'TM-Sprühen'
e38936, 'Rauhen'
e31724, 'Kleben'
e42109, 'Kombi_RC (Rauhen + Kleben)'
e48581, 'Kombi_RS (Rauhen + Stahlsohle)'
e39216, 'Beschneiden'
e38577, 'Leistenhandling'
e38824, 'Kleben Gummisohlen'

control is_one_of e10001, e10005, e10006.
constrain(control in [e10001, e10005] <==> base_type not_in [e38936]).
constrain(control in [e10001] <==> base_type in [e38534]).

Machine description

security is_one_of e10105, e10106.
constrain(security in [e10105] <==> base_type not_in [e48581]).
constrain(security in [e10106] <==> base_type in [e48581]).

optionen null_or_n_of
schafterkennung, messeinrichtung_f_stationen, takttisch_6_stat,
bearbInsel_2_stat, barcode_lesegeraet,
leistendreheinrichtung_extern, tm_sprueheinrichtung_extern.

schafterkennung is_one_of e10101, e10102, e10103, e10104.
constrain(base_type in [e39216] <==> schafterkennung not_in [e10104]).

messeinrichtung_f_stationen is_one_of e10113.
barcode_lesegeraet is_one_of e10110.

Solution approaches

My Venice homework

A simple finite domain solver.

Create the CHR-constraints from the definition of the robot.

Create an instance of a robot and create the constraints between the parts of it.

Create a list of questions from possible decisions from which a user can choose.

I'll spend a lot of time trying to create the CHR-constraints in a dynamic module to handle different instances of a specification, but I'll fail. So I decide to translate the specification symbols into a name with a leading instance prefix.

Solution approaches

Description logic

To have your own specification language may be nice but the problem exist so long that there must be something out there.

OWL and a whole bundle of languages to describe a ontology.

A least all of them are not really fit to configuration problems.

Solution approaches

ConTes / ConBaCon

ConTes: A Terminological and Constraint-Based Approach to Solve Configuration Problems.

For specification of configuration problems, they developed a formal specification language, called TRL/C.

The ConTes approach is not suitable for bigger problems.

ConBaCon: Constraint based Configuration

Both where developed in the research project VERMEIL founded by the BMBF.

Thank you.