>> SIARAS >>

Project no.:	NMP2-CT-2005-017146
Acronym:	SIARAS
Title:	Skill-based Inspection and Assembly for Reconfigurable Automation Systems
Instrument:	STREP
Priority:	IST-NMP-1

Publishable Results

Period covered: from July 2005 to October 2008 Date of preparation: 12th December 2008

Start date of project: 15th July 2005 Duration: 36 + 3 months

Project coordinator name: Dipl.-Phys. Hartmut Eigenbrod

Project coordinator organisation name: Fraunhofer Gesellschaft

Annex of "Final Plan for Using and Disseminating the Knowledge"

Publishable Results

1	**Method for representing device capabilities by means of ontologies**	**3**
1.1	The SIARAS knowledge base	3
2	**Method to query and transform ontology data**	**6**
3	**Method for re-parameterisation of devices by means of automated reasoning**	**8**
3.1	The Skill Server Reasoning framework	8
3.2	The Reconfiguration Chain	9
4	**Methods for using simulation tools in automated reasoning**	**11**
4.1	Data model and requirements for work piece representations	11
4.2	Incorporating the selected simulation tool in the skill server reasoning process	13
5	**Methods for using simulation tools to optimise process parameters**	**15**
6	**Methods to create a calibration/verification program for the on-line robot cell**	**17**

1 Method for representing device capabilities by means of ontologies

In order to enable the automatic reconfiguration of a production system, it is necessary to have a complete representation of its subsystems (devices) and their skills. The work is partitioned into different levels of abstraction. On the highest level of abstraction, general concepts of process and skill representation such as semantic web technologies are studied. On an intermediate level these concepts are elaborated and applied to specific processes and classes of devices, e.g. vision sensors or grippers. Finally, the results of the upper levels are used to instantiate and encode the skills of selected real world sensors and actuators.

Skill representation cannot be investigated without considering its counterpart, i.e. task representation. In order to investigate the connections between tasks and skills, a model for process description has to be developed. Tasks will be described in terms of prototypic processes (process templates). These process templates are made up of sequences of elementary actions, each of which is executed by a device exhibiting specific skills. A broad spectrum of real inspection and production processes has been analysed. Based on the results of this analysis, methods and tools for process and skill representation were investigated, developed, implemented and evaluated in the course of SIARAS. The focus was set on semantic web technologies as XML, OWL and RDF. Subsequent activities were geared towards the modularisation, extensibility and hierarchy of processes and skills. As a result, a set of tools for process and skill representation is available.

1.1 The SIARAS knowledge base

The ontology is one of the central concepts of the SIARAS knowledge base. It provides the structure for the underlying reasoning performed by the Skill Server framework. The ontology defines the vocabulary about devices, skills, tasks, workpieces and dependencies between these concepts.

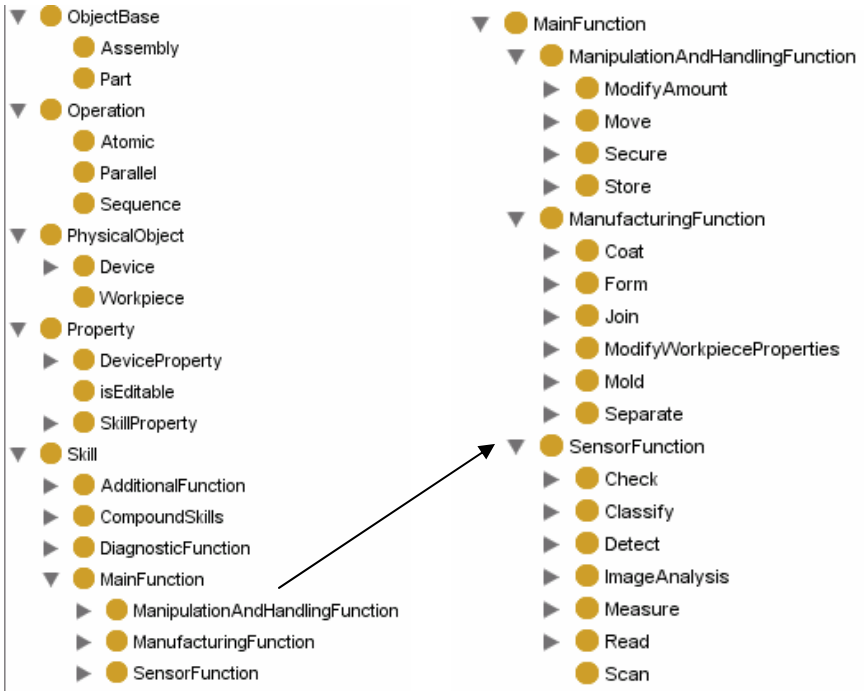

Figure 1: Top hierarchy of SIARAS ontology and second level of "MainFunctions"

On the top level, the SIARAS ontology is split into five categories.

Object Base:
Every physical object can be modelled as a simple Part, or as an assembly consisting of parts or other assemblies. It is mainly intended to hold information about geometrical dependencies of objects.

Operation:
Operations are used to describes tasks that are performed by a device. In the SIARAS knowledge base, an operation can be Atomic (an invocation of a skill), Parallel (forming a complex of at least two operations performed in parallel) or a Sequence (denoting a complex of two or more operations performed one after another

Physical Object:
Every automated work cell consists of physical objects. Some objects, Devices, are active and have skills, while other, Workpieces, are passive and are being manipulated by the devices. The device hierarchy is quite deep, with several branches, in order to mirror the diversity of actual devices used in automation systems.

Property:
The property hierarchy enumerates those properties of devices and skills which are interesting for the Skill Server to reason about it. In particular, the subtree containing the relevant parameters of devices like sensors is very numerous. Moreover, in this section are also physical parameters, communication interfaces and quality criteria.

Skill:
A Skill represents an action, that might be performed (by a device) in the context of a production process. The skill hierarchy is divided into six subcategories, each for different types of skills identified previously by the consortium. The major subcategory is MainFunction, where the manipulation, manufacturing, handling and sensorfunctions are defined.

During the project, the SIARAS consortium implemented the structure described above. It was styled for handling, manipulation and inspection processes. But due to the modular concept, it can be easily adapted and extended to other fields of applications. The following figure illustrated the structure for sensors:

After the structure of the knowledge base was ready, it had to be filled with concrete instances:

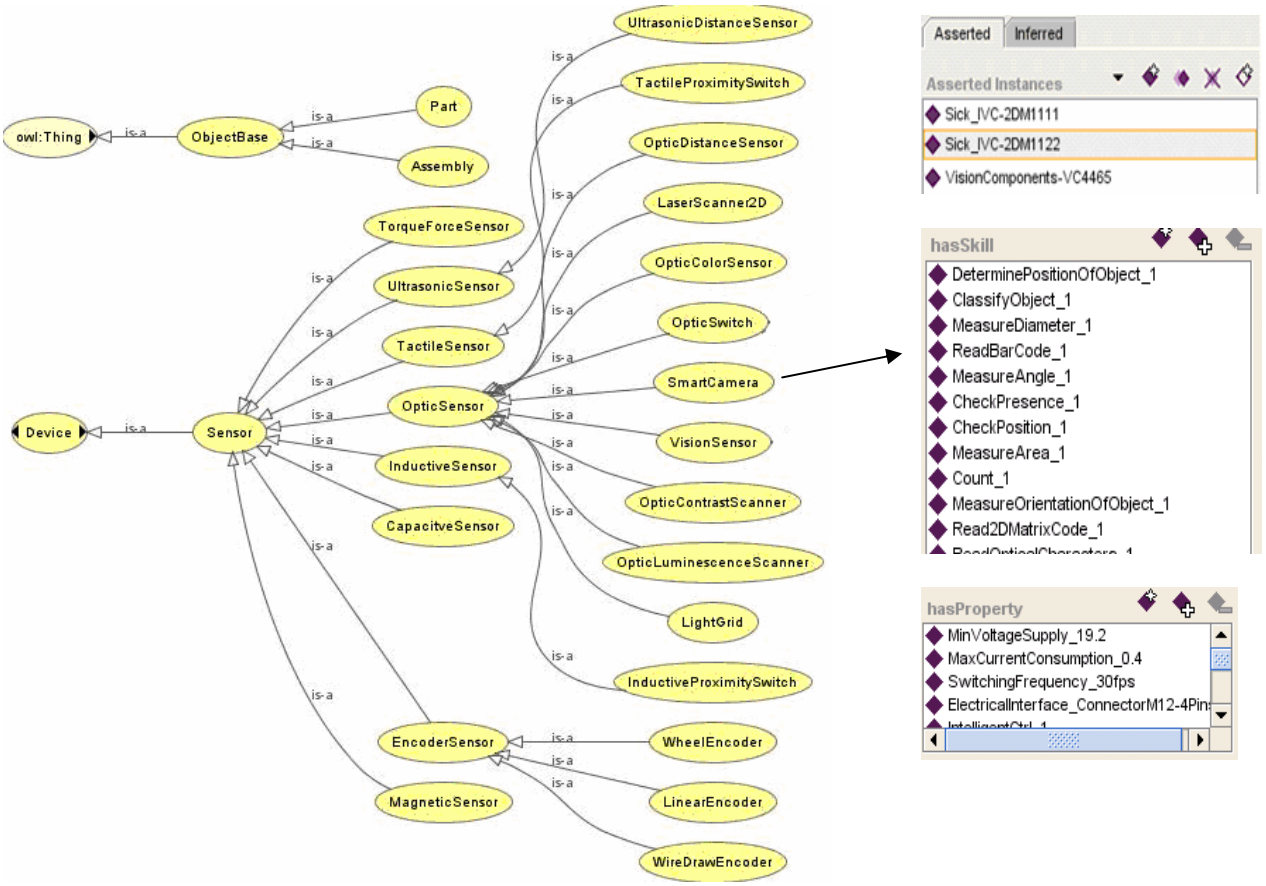

Figure 2: Structure of the hierarchy of sensors with exemplary instances

At the end of the project, the SIARAS consortium had implemented a comprehensive knowledge base for automated reasoning. The knowledge base has modular structure and about 200 devices with their skills and parameters. The devices are mainly from the fields robotic and sensors but also some manufacturing devices are stored in the knowledge base.

2 Method to query and transform ontology data

Information stored in ontologies typically requires some kind of tools to be manageable. When using the OWL format, the most widely used tool is Protégé from Stanford University. Protégé can both be used as a graphical user interface front-end to ontology, and as a Java API for accessing/manipulating the ontology. However, the API is not only cumbersome to use, the execution times required for finding types or instances in the ontology are also very long. For our ontology, the Skill Server needs about 10 seconds for complex queries. Due to even simple reconfiguration tasks needs more than 10 queries, the consortium implemented a solution to reduce this time to about 1 second. In an attempt to shorten execution times as well as trying to ease the task of accessing information from the ontology, we have been looking at using some thinking from the compiler construction area.

For reducing the execution time, the Skill Server framework needs to parse the ontology description and the various local extensions from the consortium. Thus, the framework builds an internal representation which is suitable for performing reasoning and feasibility analysis. In the other end of the Skill Server pipeline, it needs to be able to generate configurations for the control system of the devices.

One compiler construction tool that looked very promising for use in the broader area than development of programming language compilers is the JastAdd. Using the modern compiler-construction toolkit JastAdd, based on the concepts of object-oriented programming, aspect-oriented programming, and Knuth's old idea about attributed abstract syntax trees, we have implemented a tool for managing larger quantities of ontology data in an easier way than is currently possible using Protégé.

The general steps of our solution are depicted in Figure 3. We have implemented a JastAdd-based compiler that, when given an OWL ontology description as input, will generate JastAdd source code that comprises the basic parts of a compiler for ontology instances. We can then add functionality in the form of aspect modules to this compiler description before it is built. Such added functionality aspects may include anything from code for extracting information to code generators that can generate, for example, Java classes based on information extracted from ontology instances. With the new compiler, the queries are much faster (about 1 second) and the reasoning by the Skill Server can be done in an acceptable time for potential users.

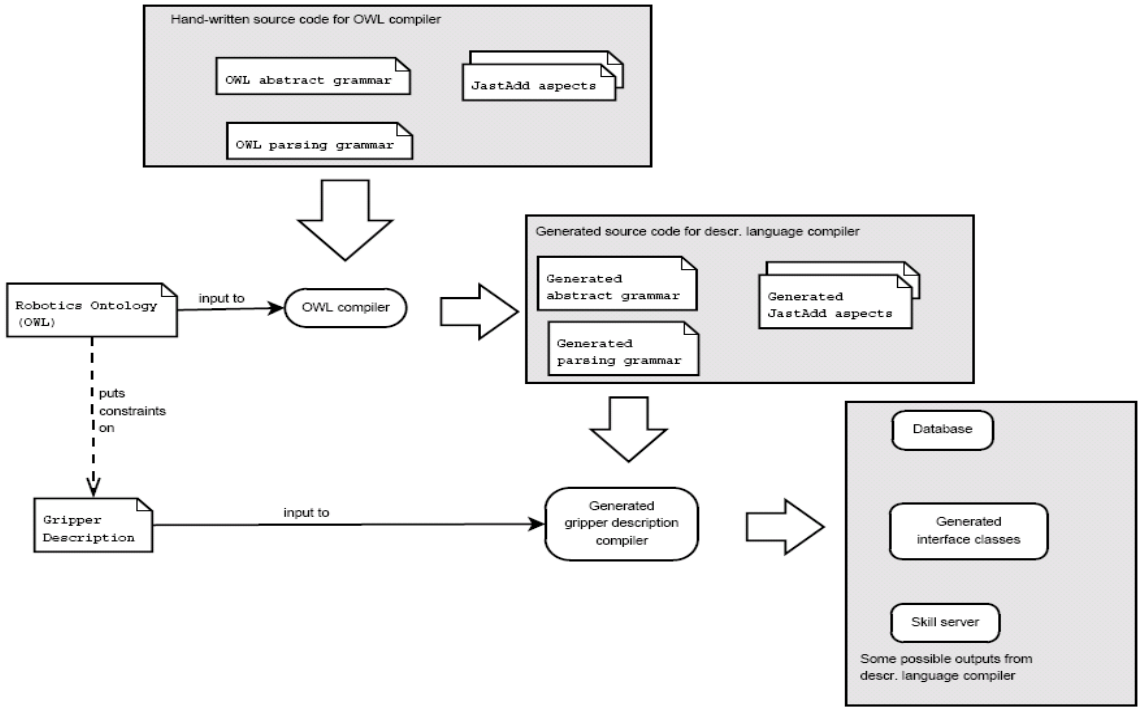

Figure 3: Compiling information manager for OWL ontologies

3 Method for re-parameterisation of devices by means of automated reasoning

3.1 The Skill Server Reasoning framework

The aim of the SIARAS project is the knowledge based automatic reconfiguration of automation systems. Therefore modules were implemented on the one hand for the representation of skills and parameters of devices and on the other hand for representation of the process flow and bounding conditions. For finding suitable configuration, a Skill Server reasoning framework with search and optimisation algorithms was developed to match the process description with the skill representation.

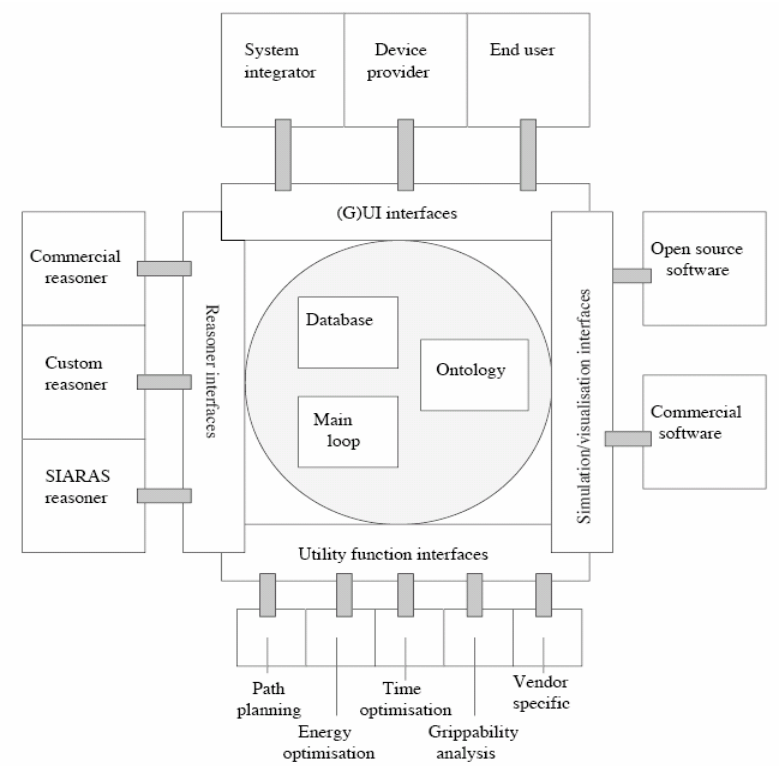

Figure 4: Principle Design of the Skill Server Framework

The reasoning frameworks works as follow:

A given task description is needed that describes the principle steps of the task and needed requirements, boundary conditions or tolerances. Therefore, the SIARAS consortium developed a Flow Chart based approach, were the user can describe and parameterize the process description step by step.

With the task description, the Skill Server is able to compare the needed requirements in the task description with his knowledge base. Therefore, search strategies were developed and implemented to perform a comprehensive search in the knowledge base. The user can limit the

search by using quality criteria (like cost or accuracy) or he can chose the devices himself by all Skill Server suggestions.

With the Skill Server reasoning framework, an automated selection and parameterisation of devices is possible to solve reconfiguration task. The consortium provides a complete framework that supports the user with an interactive and graphical reconfiguration process.

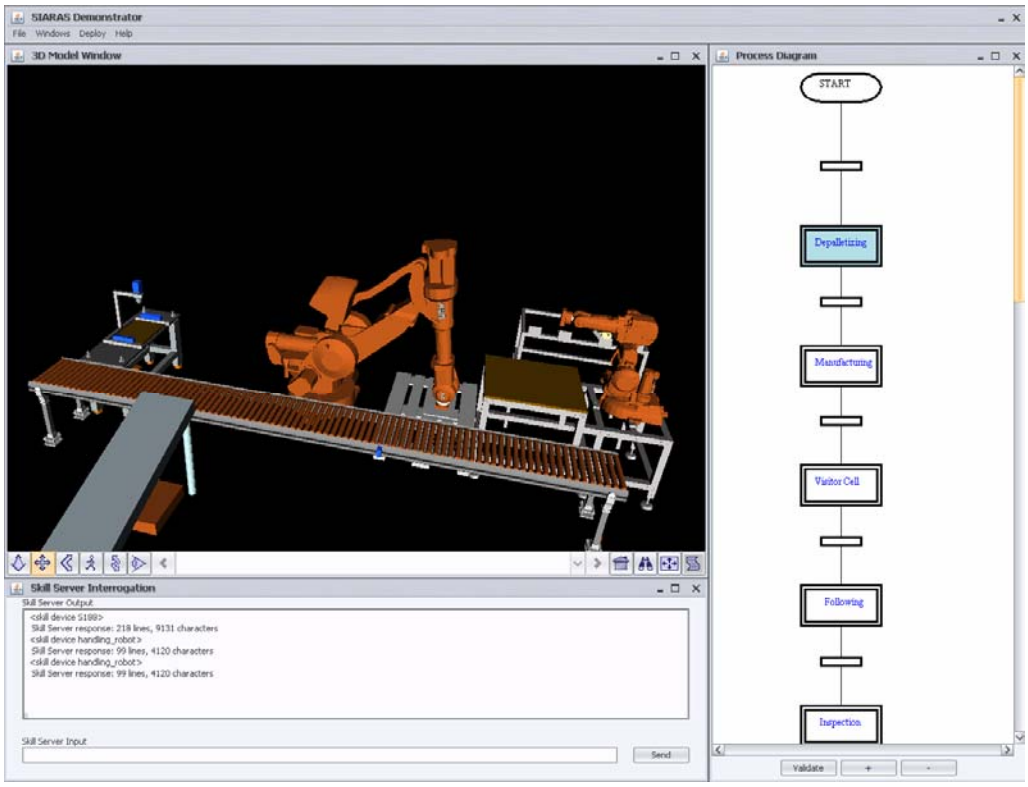

Figure 5: Grafical user interface for task descriotion and interactive reconfiguration with the Skill Server framework

3.2 The Reconfiguration Chain

It is very important for the user, that he can recognize the reconfiguration process. Thus, the SIARAS consortium developed a structured method for in interactive reconfiguration process. That means, the user changes the task description and the Skill Server shows the user step by step the need for reconfiguration and possible suggestions. To handle this in a standardised way the following process was developed by the SIARAS consortium:

Changes in the Task description:
The user can change devices, skills, device parameters or work piece properties.
Example: The integration of a new process step in the task description or a change of the workpiece material.

Check of feasibility:
The Skill Server checks if the task description is feasible. That means the Skill Server checks whether there is an appropriate device for every Skill or whether the parameters are in the given borders.
Example: The Skill Server checks, if “sensor_1” can perform the skill “detect object”.

Search for suitable devices:
If there are some skills that can not be performed by the devices in the SFC or if no device was selected, the Skill Server will search for possible devices and chose the best.
Example: For the cheapest sensor that can perform the Skill “measure diameter”.

Parameterisation of the selected devices:
When all devices are chosen, the Skill Server provides some initial parameters for the new devices.
Example: The Skill Server suggested a drilling speed for a drill bit by using the drill diameter and the material.

Optimisation of the selected devices:
In this step, the Skill Server can optimize some process parameters. This must be done with utility functions or simulation tool.
Example: With the help of simulation tools, the Skill Server optimizes the speed of the robots to get the best tradeoff of cycle time and energy consumption.

Upload results:
When all devices are chosen and reconfigured, the Skill Server must give an output.
Example: A configuration file.

4 Methods for using simulation tools in automated reasoning

This section summarizes work done within the SIARAS project on the topic of development of data models suitable for simulation tools and to study and develop data exchange schemes between the skill server and CAx (D, M, E, and PP) tools. This enables the Skill Server to utilize CAx functionality in the reasoning process, such as task planning with simulation for reachability and collision testing, and it allows CAx tools to utilize the skill server for automatic generation of new systems and configurations.

As a key component, a simulation data exchange format was defined and implemented that is suitable for the selected simulation tool. A prototype was realised that sends simulation queries to the tool enabling the use of the selected simulation tool in the skill server reasoning process.

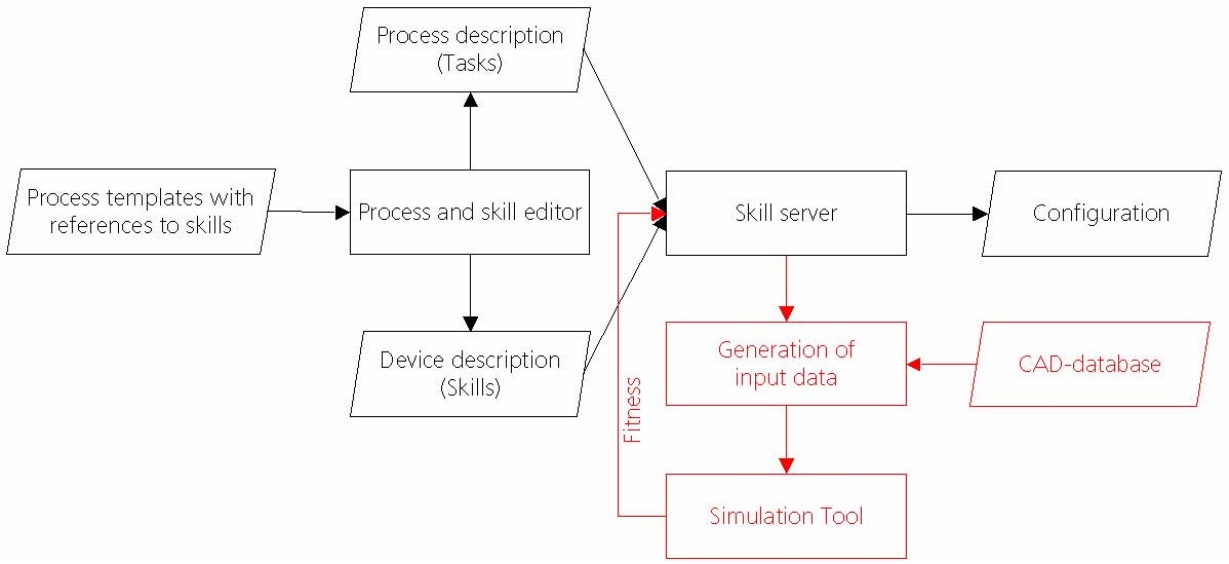

Figure 6: Simulation tools in the Skill Server process

4.1 Data model and requirements for work piece representations

To make the Skill Server able to reason on questions concerning the work piece, the information about it and the data model has to be accessible to the Skill Server. All data formats for exchange of CAx data have one main disadvantage: they are not accessible in the sense to add some meta data.

Some of the data formats offer at least the possibility to use different profiles. However, as a consequence, new parsers and tools had to be written, as the available tools do not support new profiles. Therefore, the requirements for the SIARAS Skill Server work piece representation were specified independently of the available data formats.

A work piece representation typically contains information on the following areas:

- Visualisation

This is used for the simulation of the whole process on the one hand and for adding additional information by the user on the other hand (see below).

- Physical properties

Physical properties are relevant real world features the skill server has to reason about like size, weight, material, etc.

- Dimensions

This is the information coming from the CAD tool where the work pieces were designed, i.e. the traditional work piece model.

- Accessibility

As the work pieces have to be accessed later by some device (e.g. a gripper), the accessibility of the work piece has to be defined. This includes gripping points, surface properties.

- Active scripting

Sometimes it become necessary to include some scripts within the work piece model to be able to execute some operations within a representation for example.

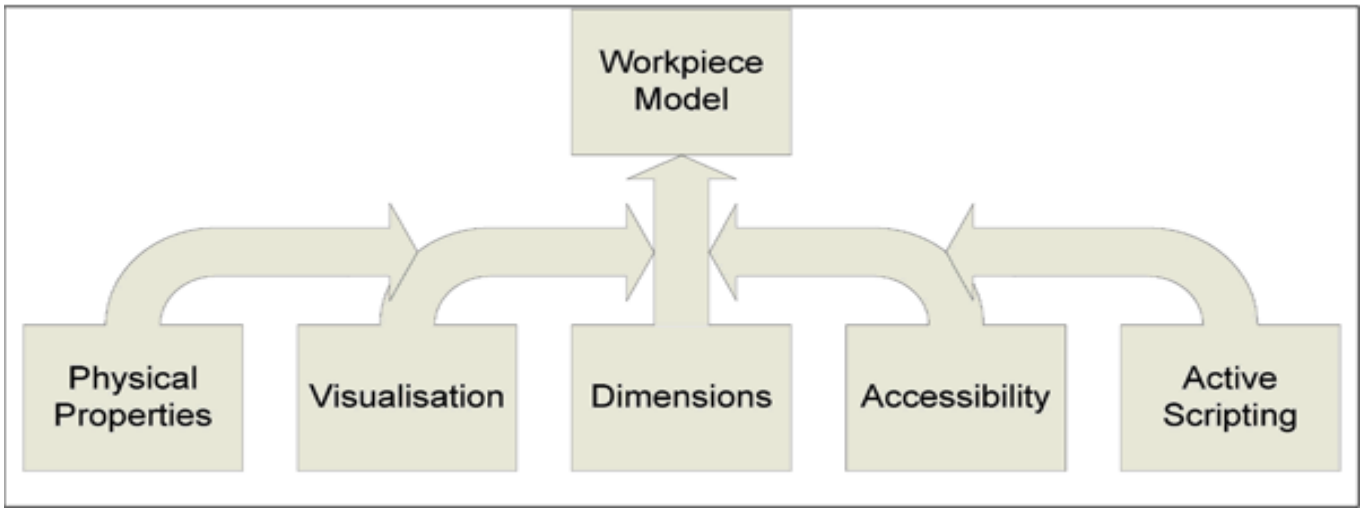

Figure 7: Information that need to be stored in a workpiece model that is used for automated reasoning

With this data model, the SIARAS consortium created a workpiece model that can be used for automatic reasoning. All relevant information can be stored as meta-data in current workpiece description of CAx data. The Skill Server can read this information and use it in the reasoning process for finding suitable devices that can handle the workpiece properties.

4.2 Incorporating the selected simulation tool in the skill server reasoning process

The framework for the integration of simulation tools in the Skill Server concept is a client-server solution that provides a general interface for formulating and asking simulation queries (here called query interface). The interface exposes a black box simulation model (here called query model) towards the skill server with only search parameters visible. A query model together with a set of parameters and a chosen question forms a query which can be asked through the query interface.

The creation of new query models is addressed to a small degree in this document. The compose interface allows the creation of simple simulation scenarios. The interface offers functionality that, within limits, can be used to create a query model corresponding to the skill server world model.

Other routes are considered but are not represented in the prototype implementation; a model can be exported from the simulation tool to the skill server. A previously created and persistently stored model can be used.

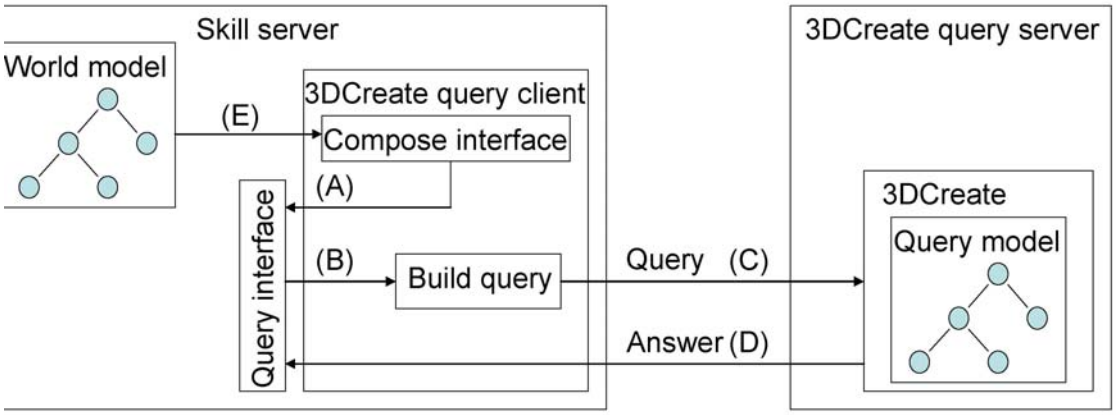

Figure 8: Architecture for incorporating the selected simulation tool into the skill server.

A simulation tool is represented in the skill server by a query client. The query client-server handles one or several questions, such as reachability test, collision test, etc. The questions that are handled can be listed through the query interface. The following is a description of the paths in the figure: (A) The compose interface is exposed through the query interface. Its basic functionality is to provide a query model. (B) The query interface forms a query from a query model, a set of parameters, and a question. This information is used by the client to build a query model for the specific parameter set and specific question suitable for the simulation tool. (C) The specific query model is transferred to the query server through a transfer mechanism. The query server loads and simulates the query model while monitoring the simulation collecting data to answer the question. (D) When the simulation ends the query server composes an answer and sends it back to the query client through a transfer mechanism. (E) The compose interface contains functionality for transforming the world model of the skill server into a query model.

Since simulation models and methods for composition may vary between tools and question types, the compose functionality is not exposed through a generic interface. The query model can be questioned for parameters and parameter ranges and populated with a parameter instance, forming a simple interface towards the skill server reasoner. Through the compose interface it is

possible to prepare a simulation covering a parameter space. This way a simulation can be prepared to accommodate a search space to be used within the skill server.

Several tools are handled by creating several client-server pairs. Each client-server pair differs with respect to the utilized simulation tool and the questions that can be answered. I.e., it is possible to have two client-server pairs for the same simulation tool, but answering different questions. It is also possible to have two different client-server pairs using different simulation tools, answering the same question. From the reasoner perspective, all tools are accessed through the generic query interface. A mechanism for finding query clients is not described in this document as such a mechanism is part of the general skill server implementation.

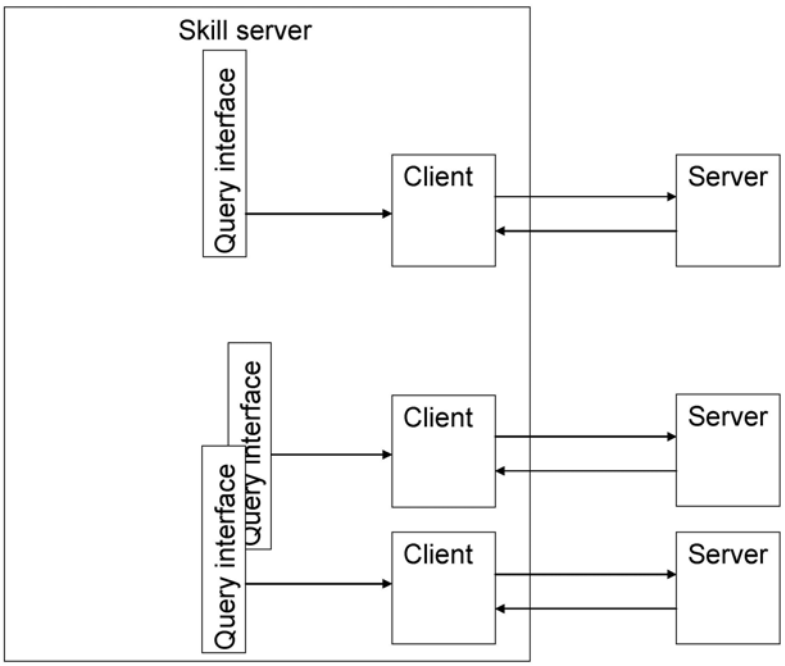

Figure 9: Query interface issued to allow generic access to several simulation tools.

The client-server pairs are built to answer specific questions. The questions that can be answered can be accessed through the query interface. To add new simulation questions to the skill server, either an existing client-server pair is updated to accommodate the new question, or a new client-server pair needs to be implemented, possibly utilizing another simulation tool.

With the developed data model and interface to simulation tools, the Skill Server framework can use the simulation tools for reasoning about geometrical issues and visualization.

5 Methods for using simulation tools to optimise process parameters

This section describes the results how to use simulation tools in the Skill Server concept for optimisation of configuration parameters. In the field of robotics some tools exist that can be used for local parameter calculation, e.g. trajectory planer for robot arms. However for global parameters like cycle times, energy consumption or the position of the devices no optimisation tools are available. Thus, during the project the consortium developed algorithms and interfaces for scanning the parameter space and optimise the process with the help of simulation tools.

To optimize the parameters of chosen devices, the simulation tool gets a set of parameters and calculates the fitness by simulating the process. For scanning the parameter space, genetic algorithms are used to generate new possible configurations. The needed fitness to evaluate the current configuration is calculated by the simulation tools until an appropriate set of parameters is found that satisfy the given constraints. Using a simulation tool as "fitness-function" there is no need for a detailed mathematical model of the work cell or the environmental dependencies.

The following function principle is used:

- Devices and basic parameters are available in the simulation tool
- User chooses the parameters to optimize
- User chooses weighted quality criteria as fitness functions

} Initial Steps

- SkillServer sets the parameters and starts the simulation
- Simulation tool calculate the needed fitness value
- Genetic algorithm toolbox varies the parameters

} Loop

- Stop with an defined abort criterion (time, number of steps…)

} End

With the implemented interface and algorithm, the Skill Server can optimise chosen configuration in a global way. Global parameters like cycle time or energy consumption can be improved. The principle was realized with the genetic algorithm toolbox of Matlab and simulation tool 3DCreate. The following image shows the principle structure of the interfaces:

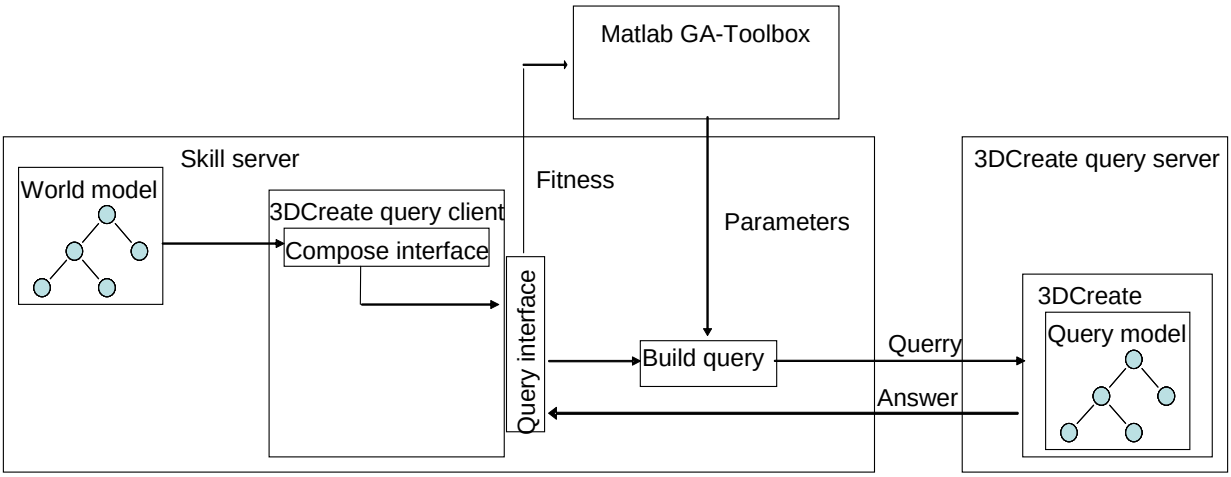

Figure 10: Interface and data exchange

6 Methods to create a calibration/verification program for the on-line robot cell

ABB has developed a tool called Robot Studio Deployment Wizard. This tool is used to react on reconfiguration suggestions from the Skill Server. The tool will make use of the Skill Server to provide setup support for process and equipment work cell changes. The Deployment Wizard will then identify these changes, generate the needed cell setup and help the user through all the steps necessary to calibrate the work object, frames, and tools used in the cell.

This is accomplished by analyzing the models, finding the reference points to be used for calibration and selecting the calibration method. This can be linked with a common control tool to make the reconfiguration process easier. In addition, 3-D snapshots are taken from the offline simulation tool, and combined with a Teach Pendant Application Program which presents the images on the teach pendant and guides the user step-by-step through the calibration process. This is combined with a robot program which can be used to quickly move the robot to all of the entered points (for calibration checks) and to save the points for future verification.