2. Showcase Industry

2.1. Concept

The main objectives pursued in the implementation of the showcase industry are the evaluation and documentation of the progress of work within the BRICS project. To achieve these goals, a realistic use case has been chosen to illustrate the strengths of the developed BRICS concepts and compare them to state-of-the-art methods.

2.1.1. Use case

When choosing an appropriate use case for the showcase industry, special importance was attached to the fact, that this showcase reflected the realistic needs of potential customers from industry. Fortunately, with the Robert Bosch GmbH an industrial partner was found, who on the one hand can point out an interesting scenario from his production line and on the other hand generously offered to provide us with feedback concerning the implementation and evaluation of our efforts.

For these reasons it was agreed to follow the vision of implementing a complete “Chaku-chaku” line using a mobile robot instead of a human operator. The “Chaku-chaku” line is an established concept for lean manufacturing at several companies like BOSCH. The concept is characterized by a circular configuration of machinery tools for consecutive assembly steps. The human worker moves along the circular arrangement and executes mainly loading and unloading tasks.

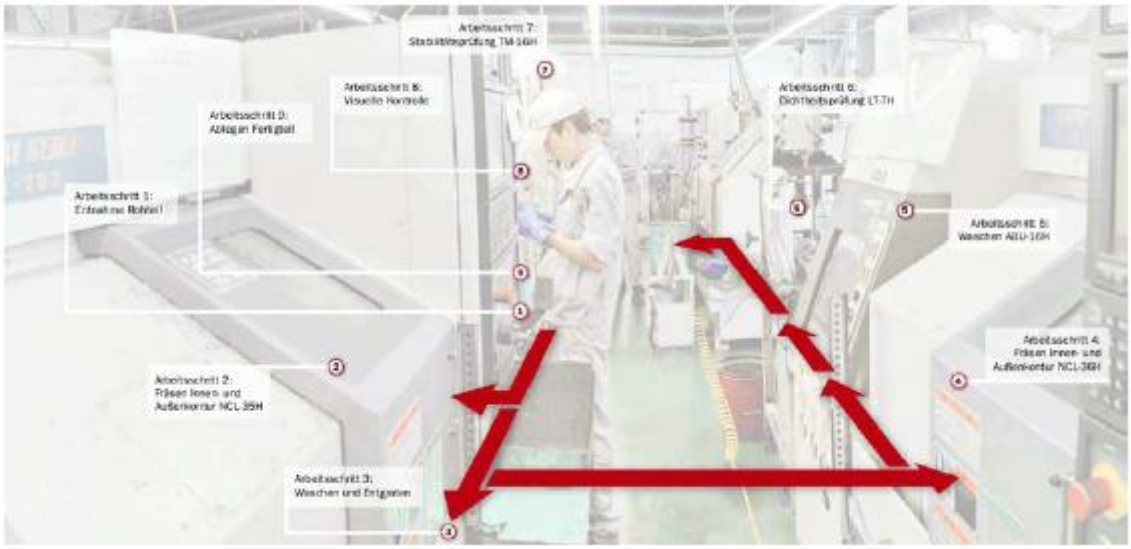

Figure 1: Chaku-chaku line as established at Robert Bosch GmbH

The vision of the showcase industry is to implement an automated version of the established concept using BRICS technologies as

- Architectural and algorithmic models
- High quality software-engineering concepts
- Hardware-components with harmonized interfaces
- Model driven engineering tool chains
- BRIDE and BROCRE
- Different access models to the robot hardware.

2.1.2. Approach

To be able to provide a relevant comparison to the current state-of-the-art, it was decided to implement two parallel versions of the use case:

a) A conventional version, implemented by a system integrator from KUKA using standard tools and software.
b) A BRICS version making use of the developed BRICS technologies available at the time of implementation.

In order to illustrate the progress of project work, three iterations with increasing complexity were planned, to be implemented in the second, third and fourth project year.

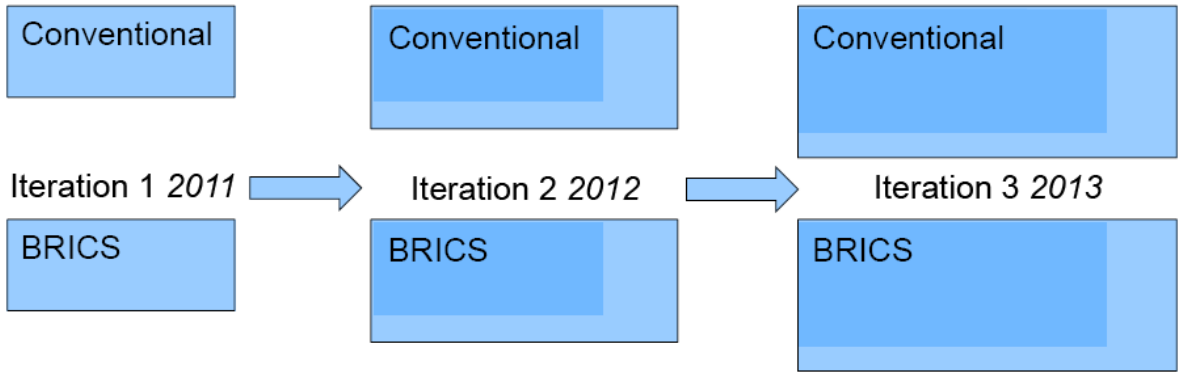

Figure 2: Implementation Plan for the showcase industry

During the implementation of the individual iterations, time measurements are taken, that will provide the background for the calibration of a software estimation model. Based on this model, a method for effort estimations will be available. Therefore time effort for the BRICS' implementations and for the conventional implementations can be compared in all three iterations. It will serve as a good estimation tool for the calculation of future industrial projects.

Iteration 1

The idea for the first iteration was to define a very basic version of the vision of a "chaku-chaku" line to ensure that the scenario can be implemented with standard tools as well as with the BRICS tools already available in the second project year. Therefore the environment was defined to be static with a non-mobile robot arm, which can easily reach three pick-up / drop-down stations arranged in a circular configuration (see Figure 3).

Figure 3: Hardware-Setup for the first iteration of the showcase industry.

Corresponding to the hardware setup, the required workflow is rather simplistic (see Figure 4). At the beginning one object is placed on each of the three transmission stations. Starting the work process, the robot is required to localize the object on station one, calculate a valid grasping position and transfer the object to the second station. There the object is to be placed next to and not on top of the object that was placed on station two beforehand. After having placed the first object at station two, the second object is to be grasped and transferred to station three and so on. The process is meant to be run in an endless loop until it is stopped by a human operator.

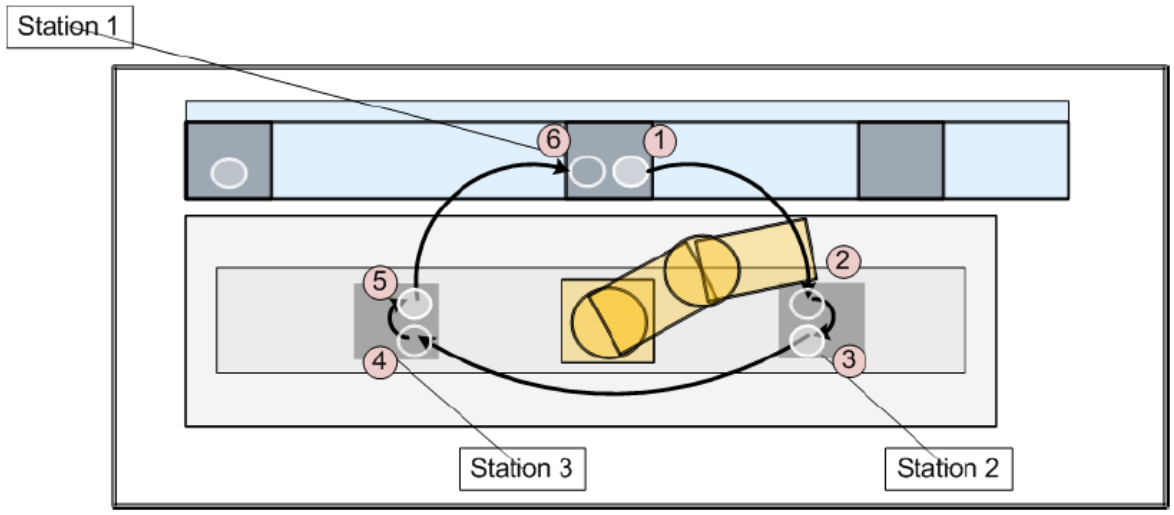

Figure 4: Workflow to be implemented in the first iteration.

To avoid putting too much effort on details of image processing, only one type of object was chosen for the first iteration. The objects can be grasped by a two-finger gripper and their white color provides a good contrast to the black plates used as transmission stations.

Figure 5: Objects used in the first iteration.

In summary, the tasks to be implemented for the first iteration of the showcase industry include:

- Loading and unloading of three working stations in a circular configuration
- System integration (sensors, actuators, …)
- Vision-based manipulation and localization
- Process integration (gripper, feeder, …)

Iteration 2 and 3

As already mentioned above, iteration 2 and 3 are thought to reflect the progress of work and will therefore each will consist of a reimplementation of the previous showcase version with a stepwise increased complexity.

For the second iteration the environment will remain static, whereas the robot will be placed on a mobile base. Further extensions in the second iteration include:

- System integration of additional components
- Path planning
- Online integration of new working stations

- Derivation from the planned behaviour has to be handled autonomously by the system.

For the third iteration it is planned to implement a dynamic environment combined with the mobile base from iteration 2. Further extensions in the third iteration include:

- Extension of the software from iteration 2
- System integration of additional components
- Moving conveyor belt
- Path planning
- Obstacle avoidance.

2.2. Implementation

The actual implementation of the showcase industry was done in two phases. In the planning phase, the participants of the different partners came together and were first confronted with the specific application, the available hardware and the requirements of the showcase. Afterwards the implementation was planned by the participants. In the second phase the showcase was implemented using as much BRICS tools and concepts as available.

2.2.1. Planning phase

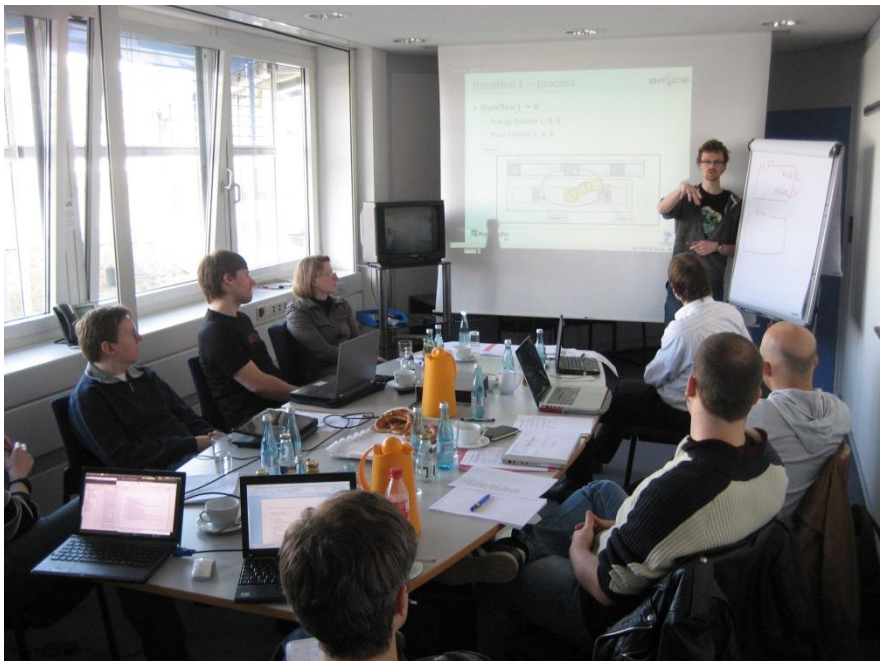

Figure 6: Participants of the evaluation workshop during the planning meeting.

Participants from BRICS work packages two, three and four were taking part in the planning phase. Based on the detailed task description given and after getting to know the available robot hardware of the system a discussion was started on how to implement the showcase. Fraunhofer IPA was listening to the discussion and taking notes on the architectural decisions and assumptions that have been made during the discussion. The participants created a component architecture describing the target implementation using the vocabulary and concepts of the developed BRICS component model. The resulting component architecture is pictured in figure 7.

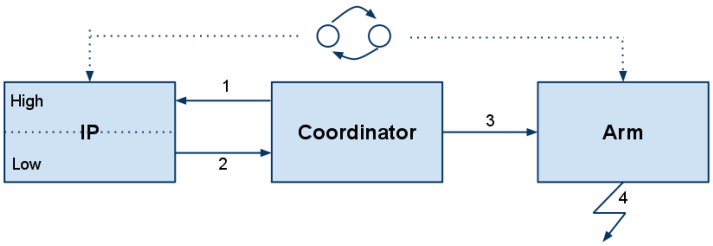

Figure 7: Planned component structure of the showcase implementation created by the participants

The architecture consists of an image processing component handling object recognition and coordinate transformation, an arm component for the Cartesian manipulator control and the control of the gripper and a coordinator component implementing the overall state machine of the system. The communication between the components was additionally specified by the participants in the planning

phase. For the exact specifications of the message types the ROS message syntax was used since it was supported by both middleware architectures that were planned to be used, namely OROCOS and ROS. The decision to use such a hybrid infrastructure was actively made by the participants to demonstrate the abstract definition of the components using the BRICS component model concept. The detailed message specifications are as followed referring to the numbers in figure 7:

1. Triggering of image processing:

   std_msgs/Bool processing

2. Result of image recognition:
   PlateStateArray of PlateStates:

   std_msgs/StringplateID

   geometry_msgs/Pose object

   geometry_msgs/Pose free

   std_msgs/BoolobjectIsFound

   std_msgs/BoolfreespaceIsAvailable

3. Target position of grasp or drop
   geometry_msgs/Pose pose
4. Finished event:
   std_msgs/Bool finished

According to this component design the necessary steps for implementing the showcase had been classified into five work packages (WPs):

- WP 1: Low-level image processing (IP.Low)
- WP 2: High-level image processing (IP.HIGH)
- WP 3: Implementation of the arm component (ARM)
- WP 4: Implementation of the coordinator component (COORD)
- WP 5: Integration and testing

To allow an efficient workflow, it was decided by the team to split up into groups of two persons each and to work on the different work packages in parallel. Considering their individual expertise and time of availability during the workshop, the following work plan was prepared during the planning phase meeting:

Tuesday	**Morning**					**Afternoon**				
GPS	Meeting					WP1	WP2	WP3	WP4	WP5
BRSU	Meeting					WP1	WP2	WP3	WP4	WP5
KUL	Meeting					WP1	WP2	WP3	WP4	WP5
Wednesday	**Morning**					**Afternoon**				
GPS	WP1	WP2	WP3	WP4	WP5	WP1	WP2	WP3	WP4	WP5
BRSU	not available					not available				
KUL	WP1	WP2	WP3	WP4	WP5	WP1	WP2	WP3	WP4	WP5
Thursday	**Morning**					**Afternoon**				
GPS	WP1	WP2	WP3	WP4	WP5	WP1	WP2	WP3	WP4	WP5
BRSU	WP1	WP2	WP3	WP4	WP5	WP1	WP2	WP3	WP4	WP5
KUL	not available					not available				
IPA	WP1	WP2	WP3	WP4	WP5	WP1	WP2	WP3	WP4	WP5

After finishing the architecture and the work plan were finished the participants were supposed to decide which BRICS developments and concept they would use for the implementation. The **BRICS component model** in its state at T25 was mainly used as a basis for the component architecture in the planning phase. Due to its abstraction over other middleware the creation of a hybrid system consisting of ROS and OROCOS was possible. For the implementation of the coordinator component and for the internal coordination of the manipulator component the state machine library **rFSM** was decided to be used. The Care-O-bot driver stack created for the showcase research RRPs was the provider for the driver of the used gripper. The control of the LBR manipulator was utilizing the **FRI** interface with an OROCOS component on the PC side.

The **OODL** library currently is mainly including laser scanner and 3D camera drivers. Since these sensors were not used in iteration 1, to reduce the complexity of the showcase, the library was not used. The model to code generator **BRICC** was mainly developed for the first versions of the BRICS component model and therefore not applicable to the component architecture that was designed; the **BRICS IDE** is still in a very early development state and was also excluded from the implementation by the participants.

2.2.2. Implementation phase

The participants were grouped into groups of two developers and started the development of the showcase in the afternoon after the planning workshop. Their work is summarized in the following showcase diary:

Tuesday 15.03.2011:

Group Walter Nowak/Ruben Smits: on Tuesday this group was integrating the gripper driver that has been taken out of the cob-driver stack into an OROCOS component, the LBR interface was chosen (FRI implementation in OROCOS) and the infrastructure to compile and develop was build up

Group Nico Hochgeschwender/Markus Klotzbuecher: The infrastructure for the rFSM state machine was built up and the coordinator state machine was planned on white board.

Group Michael Reckhaus/Sebastian Blumenthal: By reading documentation and try and error the configuration of the proprietary image recognition in the camera was discovered. The requirements for the implementation of the scenario were checked and tested.

All groups were working between 15:06 and 17:45. After a short status discussion the day was over.

Wednesday 16.03.2011:

Group Walter Nowak/Ruben Smits: After integrating the gripper and the FRI component a hardware failure in the KUKA KRC was noticed. The support of KUKA service was requested and debugging information was collected from the KRC. After about 2 hours the problem was identified to be the network card of the KRC and thus the problem could be solved. Afterwards the FRI and gripper component were tested successfully. A Cartesian trajectory controller component was integrated in the end. The work of this group was started at 9:00 in the morning. Between 11:30 and 14:27 there was a break due to lunch and the described hardware failure. Between 14:27 and 15:45 the group was working again, being interrupted by a short status discussion between 15:45 and 16:08. The group stopped working that day at 16:35.

Group Markus Klotzbuecher/Sebastian Blumenthal: After configuration of the image processing jobs in the camera a component wrapper was written interfacing the proprietary interface of the camera. The component was implemented as a ROS node publishing the 2D detection results This group also started working at 9:00, had a lunch break between 13:15 and 14:00 and participated in the status discussion between 15:45 and 16:08. The day was ended at 16:35.

Thursday 17.03.2011:

Group Walter Nowak/Alexander Bubeck: After the integration of the trajectory controller tests have been done together with the real hardware. The launching of the KUKA KRC program and the

communication trajectory controller/FRI interface was debugged. The functionality of the component was completely implemented now. The integration of a state machine using rFSM was investigated, the OROCOS Lua bindings had to be installed and understood. Glue code between the rFSM and the OROCOS infrastructure was started.

Group Sebastian Blumenthal/Michael Reckhaus: Sampling of free space based on the detected object of the camera was implemented as well as a transformation of 2D data into the 3D coordinate space.

The work of the two groups began at 11:30 and finished at 19:38. In between there were breaks between 12:24 and 12:29, between 13:03 and 13:42 and between 16:39 and 17:20.

Friday 18.03.2011:

Group Walter Nowak/Alexander Bubeck: Coordinator implementation was started based on the work of Markus Klotzbuechel and Nico Hochgeschwender on Tuesday. The integration of OROCOS Lua and rFSM was continued. The arm component was tested with regards to the integrated state machine to pick up objects at taught positions. The first communication links between coordinator and arm component were implemented and debugged.

This group worked between 09:12 and 17:52 and had breaks at 11:30 until 11:49, at 13:05 until 13:31 and at 16:48 until 16:56.

Group Sebastian Blumenthal/Michael Reckhaus: The head eye calibration was done and thus the transformation from camera coordinate system to manipulator coordinate system was implemented. The communication between image recognition and coordinator was implemented.

The work of this group was done between 9:12 and 16:00 with breaks between 11:30 and 11:49, between 13:05 and 13:31 and between 14:47 and 15:05.

Wednesday 23.03.2011:

Group Walter Nowak/Alexander Bubeck: The integration of the overall system was continued and the communication between arm and coordinator was debugged and partly rewritten because of not complying with the BRICS component model. The coordinator state machine implementation was continued.

Group Sebastian Blumenthal: Communication between coordinator and image recognition was further tested. Minor bugs in the IR code were fixed. Calibration errors and arm positioning errors were detected.

Both groups worked between 13:15 and 20:11 without breaks

Thursday 24.03.2011:

Group Walter Nowak/Alexander Bubeck: The coordinator state machine was finished interfacing both arm and image recognition. Tests have been done and the arm positioning error could be reproduced. The error was found in KRC tool configuration after changing the configuration and calibrating the KUKA LBR to the plates the positioning of the arm component was satisfied.

Group Sebastian Blumenthal: Testing of the overall system was supported. Further calibration of the system was done.

Again both groups worked between 16:10 and 20:04 without breaks

.

Friday 25.03.2011:

Group Sebastian Blumenthal: New head eye calibration of the image recognition component using an iterative approach. Finally the objects were detected reliably at the right positions.

This work was done between 13:10 and 16:00.

On Monday, 28.03.2011 the finished showcase industry was finally run. The running system fulfilled all the given requirements and was robust reacting to disturbances like moving objects during runtime. The only possible failure we noticed was in the singularity resolution of LBR. When joints reached the soft joint limits of the manipulator the redundancy resolution was not possible to fulfill the desired movements. The running system is shown in figure 8.

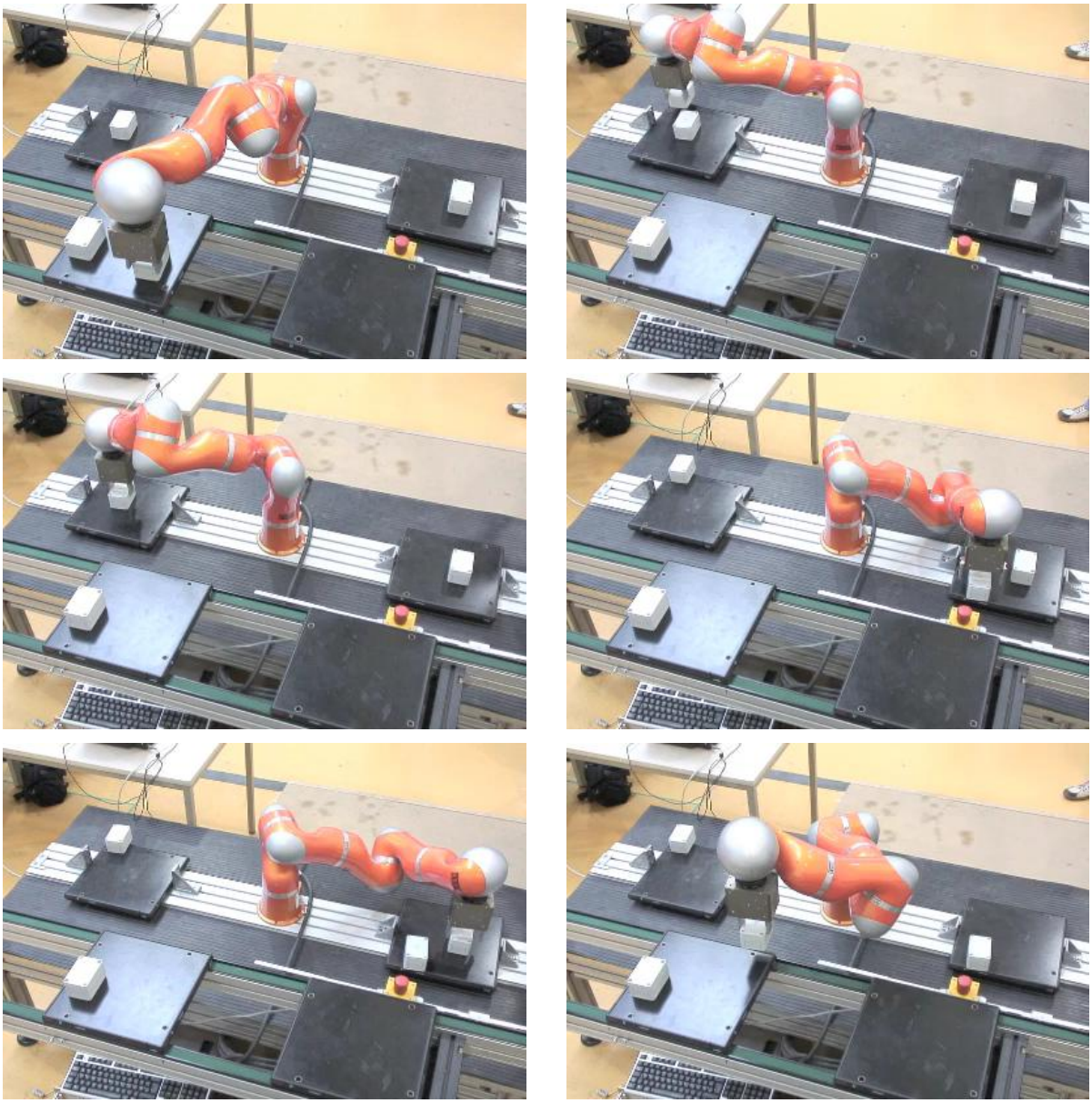

Figure 8: The working showcase industry after the implementation during the evaluation workshop

2.3. Evaluation

In order to obtain information on the software development productivity a work breakdown structure had been established based on the work documented in Deliverable D2.2 and the time spent on the different stages were tracked per person. The allocation of person hours to specific work phases allows judgement on the significance of the different software related tasks arising from implementing a service robotic scenario.

The work stages identified are:

1. Requirements and design
2. Project planning
3. Reuse acquisition
4. Component integration
5. Coding
6. System integration
7. Configuration
8. System (stress) testing
9. Maintainability testing

To allow tracking of the effort without interfering the work of the developers a tracking tool was created that allowed detailed specification of the current task the developer was working on with just two mouse clicks. The granularity of the tracking can be configured. During the evaluation workshop the tool was running with a resolution of 4 minutes. Figure 9 shows the graphical user interface of the tool as it was running on the work stations of the participants.

During the workshop only few complaints were given by the developers. While no participant felt interfered by the evaluation tool, it was sometimes not clear to the participant to what development phase an activity belongs to. These complaints will be faced by us during the planning of the next evaluation workshop.

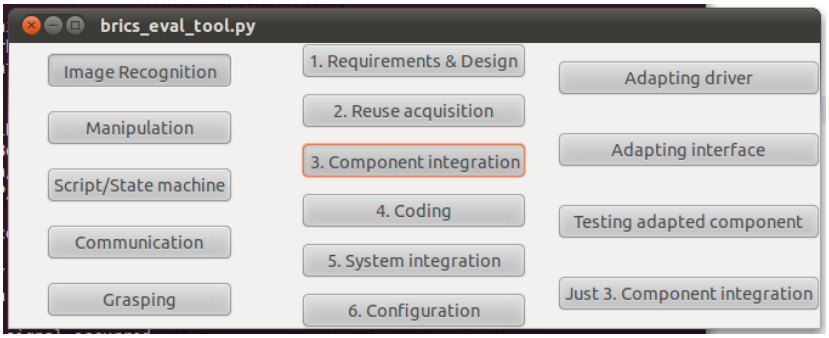

Figure 9: BRICS evaluation tool for effort tracking during the evaluation workshops

The resulting distribution of total person hours on the different phases is displayed in Figure 10.

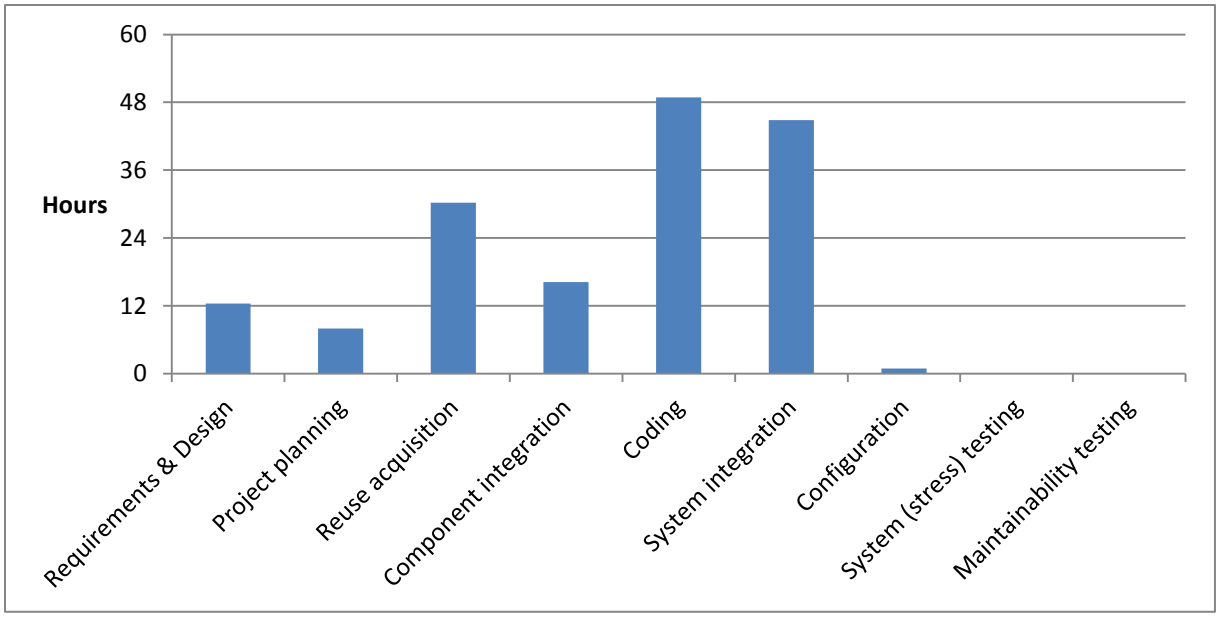

Figure 10: Distribution of person hours

In order to evaluate the software development productivity the time needed for implementation has to be related to its size and to the developers' experience.

The straight forward method of measuring the size of a software project is counting its lines of source code statements. Although this approach has been widely criticized for its considerable flaws it is common practice in software controlling as it yields at least a rough orientation of the effort put into the project.

The software implemented for the showcase industry can be divided into two major parts – code that has been proprietarily developed and code that has been reused. The own development consists mainly of four components: the arm state machine, the wrapping of the Schunk gripper driver, the coordination component and the wrapper to the Cognex insight image recognition. The overall amount of lines of code (LoC) for these components is 1510, excluding auto-generated files (such as Makefiles), blank lines and comments. The component code integrated – which needed only minuscule adjustments – consists of the control interface for the KUKA FRI and the arm trajectory generator. The overall amount of LoC for these reused parts is 2834, excluding auto-generated, blank lines and comments.

One major criticism towards counting lines of code is the massive divergence of the average amount of code needed for a specific function depending on used programming language, tools experience and many more.

In order to alleviate this problem the number of LoC was translated into a projection of function points (FPs) via a method called "backfiring"[1]. Function points are a measure of software size independent of the programming language used. They are usually counted manually which requires expert analysts but backfiring allows to transfer LoC into function points by using heuristic values of LoC:FPs ratios, one for each specific programming language.

In the showcase industry, the main programming languages were Lua, C/C++ and Python. For C, C++ and Python, heuristic ratios were already available but for Lua there was none. Thus, the average of four similar scripting languages' ratios (PHP, Python, Haskell, Perl) was derived and used for Lua backfiring.

In this way, the calculated number of function points for the proprietarily developed software part in the showcase industry is 42, the one for the integrated part is 98. In other words, roughly a 30% of the software required had to be developed from scratch whereas 70% could be realized by integrating existing software. This is already a respectable ratio of reuse that is above the average reuse ratio for

[1]Backfiring has been developed by Software Productivity Research, a US American company providing software consulting services. See http://www.spr.com/programming-languages-table.html for more information.

component based development of 40% proprietary development as opposed to 60% reuse. Future iterations of the showcase industry will reveal if the reuse percentage can be augmented even further.

Furthermore, the participants were asked to fill out a short questionnaire on their level of expertise regarding service robots and software development. The aim of this practice was to gain knowledge on the overall team experience so the person times measured can be related to the developers' level of proficiency.

In the questionnaire, the developers had to judge their own level of expertise using a Likert scale from 1 to 5 (1=novice, 2=moderate, 3=intermediate, 4=advanced, 5 =expert). The distribution of the questionnaire results is displayed in figure 11.

The average of the team working on the showcase industry, using 1 as the lowest possible score and 5 being the highest, is approximately 3.7 which is an indication that the team is skilled, with a tendency towards advanced experience.

On the whole, approximately 166.5 person hours were required to implement the software for the showcase industry. Statistics of the influence of team experience suggest that the maximum positive influence of the team experience on the overall software productivity is 55% (compared to a team with average (=3) experience), the worst influence being -87%.

Interpolating from these values, a team with average experience, i.e. slightly less experienced team than the given team for the showcase industry, would need approximately 199.1 person hours for this project.

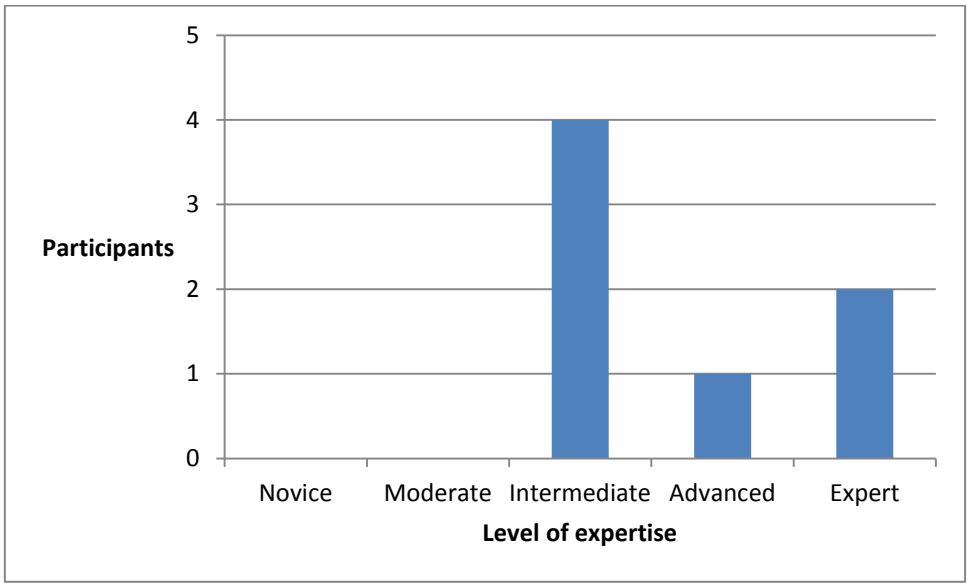

Figure 11: Distribution of team's level of expertise

In order to assess the validity of these self-assessments the attendees were asked to give detail on the time spent with programming and with robots. Although there are large fluctuations the overall picture seems plausible as all of the team members' total time spent with robots and programming ranged from between 2 and 10 years.

Assuming that the data collected is representative it forms a basis for extrapolating and estimating the time frame of similar future software projects, allowing for experience-related adjustments. It will be particularly interesting to see how future iterations of the BRICS showcase will perform.

2.4. Implementation by KUKA Systems

As introduced in section 2.1.2 the approach for evaluation was designed with two implementations of the showcase industry. In addition to the BRICS implementation analysed above the implementation with conventional tool chains and concepts was done by KUKA Systems. The application requirements, the hardware setup and the target application were completely equal to the implementation by the BRICS developers. Similar to this first implementation the execution was divided into two phases, the planning phase and the implementation phase. The actual evaluation using the same tools and concepts as in the BRICS evaluation was done only on the implementation phase.

Between the 28th of April and May 3rd the KUKA developer Andreas Koeglmeier was implementing the application at Fraunhofer IPA. The implementation was done completely on the KUKA KRC controller. The image recognition of the Cognex insight camera was integrated in the KRC by the KUKA vision software, the gripper was also directly commanded by the KRC. The main work that had to be done was configuration of the different products to work with each other and the hand-eye-calibration of the system. Figure 12 shows the developer while calibrating during the evaluation workshop.

Figure 12: KUKA developer during calibration of the showcase industry system

During the evaluation workshop the KUKA developer was using the Fraunhofer IPA tracking tool to record the distribution of hours needed in the development phase's described in section 2.3. This distribution is displayed in figure 13.

Obviously, the single expert needed far less time than the academic team to fulfil the task – 17 hours 43 minutes as opposed to 161 hours 25 minutes. There are multiple reasons we see for this effort difference. The time for reuse acquisition (30 hours in the BRICS case vs. 3.5 hours by KUKA) was barely required by the KUKA developer since he only had the KUKA tool chain available. The expertise of the developer in comparison to the BRICS participants was much higher with regards to the tools he used. The further development of BRIDE and BROCRE will reduce these factors in the case of the BRICS implementation.

The downside of the KUKA approach is the strong dependency on the used hardware and specific setup of the environment. During the conception phase of the showcase multiple changes had to be made to the specification and the hardware setup to fulfil the needs of the KUKA tool chain. As the complexity of the project is rather small these demands by KUKA could be handled; in a larger project, the advantages of flexibility probably would outweigh the shortcomings mentioned above.

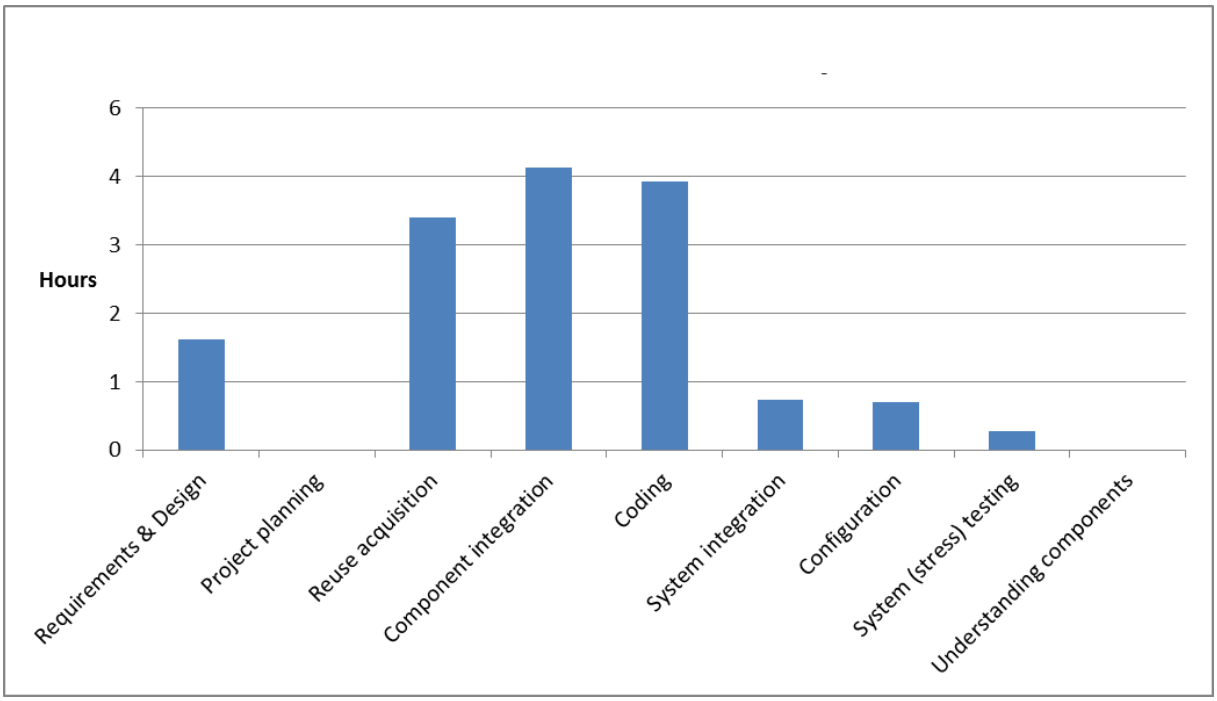

Figure 13: Effort distribution of KUKA developer during implementation of the showcase industry

.

2.5. Results and conclusions

The main purpose of the showcase industry and the first evaluation workshop was to demonstrate the current BRICS developments on a realistic robot system and to evaluate the progress of the project towards its goals. The targeted audience of the showcase industry are industrial application developers, system integrators and manufacturing companies. Therefore an assembly line at BOSCH was chosen as a realistic scenario.

In comparison with KUKA Systems the effort of the BRICS implementation has been significantly higher, while being more flexible with regards to the hardware and scenario. In the evaluation of the showcase it was shown that most effort was spend in the phase “Reuse acquisition”, “Coding” and “System integration”. These fields have the biggest potential for improvement and should therefore be in focus for the development of BRIDE, BROCRE and the BRICS model driven engineering tool chain.

The detailed requirements for the next iteration of the showcase are:

Calibration:

Compared to the KUKA implementation there was no best practice guideline and no tool support for calibration of camera systems towards the robot available. Hence a noticeable amount of time was spent to manually iteratively calibrate the image recognition results to the manipulator movements. The BRICS wiki-book could be a good place for such guidelines; an evaluation of available calibration tools will additionally improve the development process in the next iteration.

Component generation:

The BRICS component model was very helpful especially in the planning phase and made it possible to easily integrate different communication infrastructures in the scenario. But since the component model has no formal specification yet, validating the created code against the model was not yet possible. Also a reasonable amount of time was spent with implementing the component wrappers for the different used components. This task can be automated once a formal specification of the component model is available and model to code tools can be created. Such tool would have reduced the effort in the phase “coding” a lot.

BRICS-MM:

The main reason the BRICS mobile manipulation planner library was not used during the evaluation workshop was a lack of a path interface on the joint level with interpolation in the FRI component. This interface should be provided and tested until the next evaluation workshop to analyse the BRICS-MM library in the showcase industry setting.

Coordination of components and the overall system:

For the coordination of the manipulator component as well as for the overall system the rFSM state machine library was successfully used. The actual state machine code was written with low effort and the documentation was valued good by the developers. But the integration of the state machine with the actual robot system middleware was more difficult and less documented. Few examples were given and thus the glue code between the state machine and the rest of the system is large compared to the functionality and difficult to be reused.

Repository of components:

The current BROCRE repository on github.com was barely used during the evaluation workshop. Since the necessary components of the system are now known they should be added to the repository. Integration with the evaluation infrastructure of WP2 can also reduce the effort in the “reuse acquisition” phase. Equally the documentation of the tools and components used was spread on different resources at the time of the evaluation workshop.

Integration of proprietary components:

As common in most robot installations for industrial audience’s proprietary components were used with proprietary interfaces. In the case of the showcase industry the Cognex Insight object recognition and the KUKA KRC were closed components with own interfaces and internal states. From the BRICS component model and from the tool chain there should be support and documentation on the handling of these communication and computation patterns to enhance transparency on the impact of the overall system. To give an example, the singularity handling of the KRC had a not modelled impact on the overall coordination of the system in the implementation of this showcase.

A bias especially in contrast to the KUKA Systems implementation we noticed was the fact that all components of the BRICS implementation had to be coded during the evaluation workshop. This bounded developer time while giving us fewer possibilities to introduce aspects in the workshop that allow further and deeper evaluation. Hence the necessary functionality will be given to the participating developers in advance giving them the possibility to implement the components before the evaluation.

Since the recommendations and results of the evaluation workshop have a very granular appearance it makes sense to increase the number of evaluation workshops in shorter time intervals. This would also allow an evaluation of specific aspects of the BRICS concepts and tools.

The analyses of the produced code confronted with the effort of the developer to develop the code, as known in software engineering, was the methodology used in the first evaluation workshop on the showcase industry. The results of the evaluation gave us the possibility to point out specific flaws and benefits of the current state of the BRICS project. Therefor we will continue to use this approach in the coming evaluation workshops.