


# Knowledge-Based Instrumentation and Control for Competitive Industry-Inspired Robotic Domains

Tim Niemueller<sup>1</sup>  · Sebastian Zug<sup>2</sup> · Sven Schneider<sup>3</sup> · Ulrich Karras<sup>4</sup>

Received: 4 May 2016 / Accepted: 9 July 2016 / Published online: 6 August 2016  
© Springer-Verlag Berlin Heidelberg 2016

**Abstract** Autonomy is an increasing trend in manufacturing industries. Several industry-inspired robotic competitions have been established in recent years to provide testbeds of comprehensible size. In this paper, we describe a knowledge-based instrumentation and control framework used in several of these competitions. It is implemented using a rule-based production system and creates the task goals for autonomous mobile robots. It controls the environment's agency using sensor data from processing stations and instructs proper reactions. The monitoring and collection of various data allows for an effective instrumentation of the competitions for evaluation purposes. The goal is to achieve automated runs with no or as little human intervention as possible which would allow for more and longer lasting runs. It provides a general framework

adaptable to suit many scenarios and is an interesting test case for knowledge-based systems in an industry-inspired setting.

**Keywords** Mobile robotics · Autonomy · Rule-based production systems · Smart factory · Factory instrumentation · RoboCup industrial · Benchmarking · Industry 4.0

## 1 Introduction

Industrial production requires a number of automated systems on various levels that instruct, control, and monitor the involved processes. **These range from low-level real-time control of individual machines to enterprise resource planning on a strategic level.** Many of these systems in various industrial sectors have achieved a high degree of *automation*. Movements like *Industry 4.0* strive to go even further and to build networks of manufacturing resources like machinery and robots that are *autonomous*, capable of controlling themselves in response to different situations, and that also incorporate the relevant planning and management systems [13]. In this effort, *smart factories* are considered a key component. These are context-aware facilities in which manufacturing steps are considered as services that can be combined efficiently in (almost) arbitrary ways allowing for the production of various product types and variants cost-effectively even in small lot sizes, rather than the more traditional chains of mass production facilities [23]. Such factories require very dynamic and diverse production workflows, moving workpieces efficiently along various routes with changing production steps and handling a variety of production machines and objects. This involves multi-robot task-level reasoning, planning,

---

T. Niemueller was supported by the German National Science Foundation (DFG) research unit *FOR 1513* on Hybrid Reasoning for Intelligent Systems (<http://www.hybrid-reasoning.org>).

---

✉ Tim Niemueller  
[niemueller@kbsg.rwth-aachen.de](mailto:niemueller@kbsg.rwth-aachen.de)  
  
Sebastian Zug  
[zug@ovgu.de](mailto:zug@ovgu.de)  
  
Sven Schneider  
[sven.schneider@h-brs.de](mailto:sven.schneider@h-brs.de)  
  
Ulrich Karras  
[ulrich-karras@t-online.de](mailto:ulrich-karras@t-online.de)

<sup>1</sup> Knowledge-Based Systems Group, RWTH Aachen University, Aachen, Germany

<sup>2</sup> Otto-von-Guericke University, Magdeburg, Germany

<sup>3</sup> Bonn-Rhein-Sieg UoAS, St. Augustin, Germany

<sup>4</sup> RoboCup Executive Committee, <http://www.robocup-logistics.org>

and scheduling, and versatile object perception and mobile manipulation.

To develop such systems, testbeds of comprehensible and manageable sizes are required to create appropriate methods and techniques and determine their effectiveness. The RoboCup Industrial Logistics and @Work Leagues (RCLL and @Work) are two industry-inspired application-oriented efforts under the RoboCup umbrella. In a smart factory manufacturing context the RCLL focuses on automated multi-robot reasoning and planning, and @Work deals in particular with object perception and mobile manipulation. The RoCKIn and RockEU2 projects implement specific functional and task benchmarks for such applications. In these scenarios, robots have to coordinate and operate a number of physical processing stations according to dynamic orders and perform complex manipulation operations. In an industrial scenario, *instrumentation* is the process of monitoring the execution of the tasks and evaluation with suitable performance metrics. An interesting observation is that in industrial contexts, the environment itself has agency and requires *control* through instructing actuated entities based on a flow of sensor data. In the aforementioned scenarios these crucial tasks are performed by the referee box (refbox). It determines and posts dynamic orders to robots, instructs, controls, and monitors processing stations, and records relevant data for evaluation. In the industrial motivation, such a system is akin in parts to manufacturing execution systems (MES) and supervisory control and data acquisition (SCADA) components. In most other RoboCup leagues, like robot soccer, a human is (still) refereeing the game and using the refbox merely to communicate with the robots. However, *we strive for refbox autonomy to the largest extent possible*.

In this paper, we describe in detail the RoboCup Industrial approach to autonomous refereeing. The general approach is based on a knowledge-based modeling and execution system, which was developed in the RCLL since 2013. The system was then extended and adopted in @Work. Both systems share the same core with encodings for the respective domains in a rule-based formalism. The differences are mostly in the frontends towards the user (different modes of operation require specialized interfaces) and the robots (different communication infrastructures are currently evaluated). Especially in the RCLL, the refbox has achieved a high degree of autonomy. It is also very well integrated with a full simulation of the environment [32] providing the same environment agency in simulation and the real world. Another important aspect is the acquisition and analysis of data to benchmark the respective key performance indicators.

In Sect. 2 we introduce the scenarios, followed by a requirements analysis and an industrial grounding in Sect. 3. The knowledge-based core is detailed in Sect. 4.

We highlight some related work in Sect. 5 and conclude in Sect. 6.

## 2 Industry-Inspired Robotic Scenarios

With ever more autonomous mobile robots entering factory floors for production according testbeds have been established. RoboCup [15] is the best-known international initiative to foster research in the field of robotics and artificial intelligence through competitions. In 2016, the RoboCup Industrial umbrella league was established as an effort to foster the cooperation of industry-inspired leagues such as the Logistics League and @Work. Furthermore, the RoCKIn EU project focused on suitable benchmarks. In the following, we describe these scenarios that share the same refbox.

### 2.1 RoboCup Industrial Logistics League

The RoboCup Industrial Logistics League<sup>1</sup> (RCLL) tackles the problem of production logistics in a smart factory. Groups of three robots have to plan, execute, and optimize the material flow and deliver products according to dynamic orders in a simplified factory (Fig. 1). The challenge consists of creating and adjusting a production plan and coordinate the group [18]. In 2013, *the refbox was introduced as a simplified Manufacturing Execution System (MES) with the goal to run games fully autonomously [22]*, in particular due to the high complexity for human referees.

A game is split into two major *phases*. *In the exploration phase, the robots must determine the positions of their teams' machines and recognize and report markers and light signal states*. During the *production phase, a production process must be maintained and optimized according to dynamic order schedules which are announced to the robots only at run-time*. The robots must coordinate for an efficient production workflow fully autonomously.

Therefore, the RCLL focuses on the topics of automated planning and scheduling, reasoning under uncertainty, and multi-robot cooperation. Other robotics aspects are intentionally kept simpler, e.g., handling of the machines or perception. The planning is open to a variety of approaches, from local-scope (single robot) to global-scope (overall fleet) planning, to distributed and centralized approaches [21]. A capable simulation of the environment is available as open source software to further corroborate this focus [32]. The RCLL task and its simulation also

<sup>1</sup> <http://www.robocup-logistics.org>.



**Fig. 1** Teams carologistics (robots with additional laptop) and solidus (pink parts) during the RCLL finals at RoboCup 2015 (Hefei, China) (color figure online)

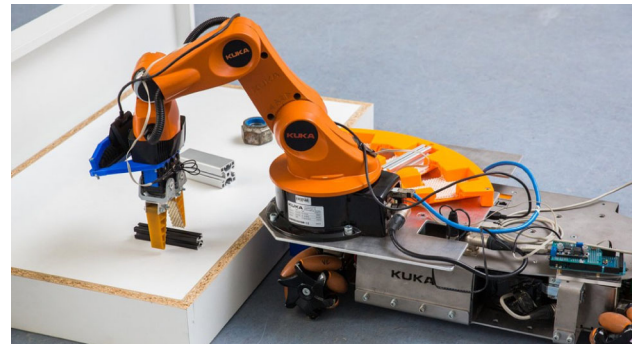
form the foundation for a Robot Planning Competition Tutorial at ICAPS 2016 [19].

## 2.2 RoboCup Industrial @Work League

The RoboCup Industrial @Work League<sup>2</sup> (short @Work) was established as a demonstration league in 2012 and is implemented as an official competition since 2014. Currently the rulebook defines nine challenges inspired by industrial mobile manipulation and transport tasks. A competition combines these tasks in separate runs with an increasing complexity. The lowest level evaluates the autonomous navigation capabilities of the system, the second addresses object recognition and grasping. The highest ranked challenges integrate up to 13 object types that have to be taken from small workspaces, a conveyor belt, or a rotating table. The manipulation objects are motivated by industrial scenarios and represent a challenging selection of materials and shapes (profiles, nuts, screws). Each object has to be transported to a new position and dropped there carefully or be placed into object-specific cavities. Each team is free to generate a schedule that optimally orders the individual operations. The referees count the number of correct delivered objects. Additionally, the team with the fastest run receives bonus points. Penalties are given for collisions with the environment and handling of incorrect objects. Figure 2 shows a robot grasping an object. The scenario is similar to the one depicted in Fig. 3.

## 2.3 RoCKIn and RockEU2

Robot Competitions Kick Innovation in Cognitive Systems and Robotics (RoCKIn) [1] was an EU-funded project



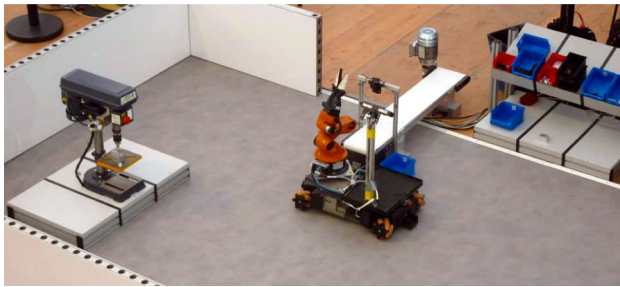
**Fig. 2** @Work robot of the robOTTO team in front of a rack grasping an object. The robot is equipped with a 3D sensor close to the gripper and two laser scanners in the front and on the backside

in the Framework Programme 7, running from 2013 to 2015. Inspired by RoboCup, one aspect is to perform in an industrial context [7]. This competitions as well as their software and hardware setup are continued in the currently running RockEU2 project, funded under the EU Horizon 2020 Framework Programme.<sup>3</sup>

The objective of RoCKIn is to raise the state of the art in industrial, manipulation-oriented competitions by combining scientific evaluations and industry-relevant aspects. To achieve these goals, the competitions are designed as reproducible benchmarks focused on the analysis of software systems (by standardizing to a common hardware platform). Functionality benchmarks (FBM) focus on the evaluation of one specific capability, such as object localization or manipulation, while task benchmarks (TBM) evaluate the integrated software in a more complex scenario including navigation, perception and manipulation. The industrial influence is reflected in the selected types of benchmarks which are set in a robot-assembly scenario. Consequently, the objects to be handled by the robots are real parts. To reflect these choices also naming was identified as an important aspect, especially to approach industrial stakeholders. For instance, the teams participate in (scientific) *benchmarks*, not *games* (like in RoboCup). Similarly, to highlight the MES aspect, the refbox is labeled *Central Factory Hub* (CFH) in RoCKIn. A further step is the addition of active, networked devices in the testbed with which the robots must interact. These devices include a conveyor belt, a drilling machine and a force-fitting machine with a quality control camera (Fig. 3). The CFH is the central instance for communicating with the robots, the devices and the benchmarking infrastructure [28]. Reusing the field-tested RCLL refbox proved to be a valuable approach since more time could be spent on designing and implementing the domain-specific logic

<sup>2</sup> <http://www.robocupatwork.org/>.

<sup>3</sup> RoCKIn and RockEU2 also have domestic service robot branches, which are not further detailed for the sake of brevity.



**Fig. 3** A robot interacting with the active RoCKIn@Work testbed. The visible devices include the drilling machine and the conveyor belt with the quality control camera

instead of designing and debugging the reasoning engine or communication protocols.

### 3 Refbox Requirements and Industrial Grounding

Each robotic competition requires a set of rules and regulations specifying the environment, the tasks to accomplish, execution constraints, and the scoring mechanism. In RoboCup, these are defined in rulebooks. They declare the scoring mechanisms, e.g., for shooting a goal in robot soccer, recognizing objects, manipulating the environment, orderschedule constraints, and likewise penalties for collisions, missed objects, or wrong interactions. The interpretation and application of these rules is in hands of referees. Most of the time, these are humans observing the game and call out rewards or penalties for misconduct. Application-oriented leagues like RoboCup@Home, Rescue, or the RCLL and @Work increase the complexity tremendously. Especially in the industrial context, the environment itself acts as an agent, monitoring and instructing processing stations. It also has several variables, e.g., the exact positioning of machines or barriers, which must be determined randomly for each game. Consequently, the referees additionally have to determine the task goals and prepare the environment according to that random assignment.

The presented industry-inspired scenarios pose high demands on human referees. A large number of referees is required to cover the competition arena and to be able to observe each relevant interaction, especially in a multi-robot scenario like the RCLL. In @Work, each referee maintains a log of observations which is aggregated and evaluated after the game, possibly incurring inconsistencies and making it hard for visitors to follow the game. Having a human in the loop also adds a subjective element to the evaluation. Applying performance metrics is virtually impossible without a sufficient amount of data (which human referees typically cannot provide).

Therefore, the environment agency, variability of games, and the task complexity hard to fully observe by human referees calls for an autonomous entity as referee—the referee box. In the following, we outline the resulting requirements in more detail and provide an industrial grounding, how the structure of the refbox fits into an industrial setting.

#### 3.1 Requirements

To provide an effective, transparent, and objective evaluation system, the referee functionality should be automated. In the following, we outline the three main requirements.

*Agency and Control* With an environment with its own agency in the presented industry-inspired scenario, the refbox needs to provide this agency in that it processes incoming data from processing stations and instructs them appropriately. With additional sensors like an overhead camera system [17], specific situations like robot collisions could be determined automatically.

The refbox must also communicate with the robots in the arena to publish task goals like orders, transport requests, or manipulation tasks, or to receive feedback like reports or robot intentions and positions. This requires a suitable communication infrastructure, a defined set of messages, and possibly encryption parameters.

*Instrumentation and Benchmarking* The basic functionality regarding instrumentation is the determination of basic evaluation criteria, i.e., scoring. The refbox must recognize correct (shooting a goal, lifting an manipulation object, delivering a product, or localization of a victim) and incorrect behaviors (collision, wrong positioning, foul play) relevant to the rules. A complete monitoring is mostly infeasible due to the high demands in terms of sensors and processing capabilities. For example, the detection of a soccer goal can be detected with reasonable effort and expense while a reliable collision detection for arm manipulation requires a large number of sensors. Therefore, we do not necessarily propose a complete automation of the evaluation, but rather to cover all facts relevant to given performance criteria.

Benchmark tests offer an additional opportunity to evaluate individual runs as well as a whole competition [2]. A refbox that provides environment agency is already in possession of a wealth of data for performance evaluation. Additionally, logging (complex) internal state updates allows to trace games in detail. Integrating external sensors or separate benchmark infrastructure into the refbox could provide for a unified evaluation. Identifying relevant performance metrics, e.g., by adapting existing industry key performance indicators to the robot



scenario as a holistic benchmark [23] allows for a fine-grained analysis.

**Visualization** In order to make it easier to follow a game, for participants and visitors alike, a comprehensible visualization of the state of the game is desirable. It should contain general information about teams, schedules, and the current scoring. Furthermore, during a run relevant information like an environment map, robot positions, and intentions should be displayed automatically. Additional cameras or other sensors may provide more interesting details not visible to visitors directly.

### 3.2 Industrial Grounding

Modern factories usually follow a specific structure for their IT infrastructure based on the layer architecture outlined in the first part of the standard IEC 62264 [11]. The layering, often called the “automation pyramid” is depicted in Fig. 4. It defines a generic architecture covering the different levels of computerized manufacturing systems (top–down) as follows.

- L4: enterprise resource planning (ERP) as the top-level strategic and supply management.
- L3: manufacturing execution systems (MES) to track and document the transformation of raw materials to finished goods and to instruct and monitor machines.
- L2: supervisory control and data acquisition systems (SCADA) that provide remote access to specific machine processes.
- L1: automated process control often based on programmable logic controllers (PLC) to control a machine.
- L0: the physical process.

This standard matches the requirements of the envisioned smart factory [26]. Therefore, the referee box has the additional effect that it fits into this architecture at the MES and SCADA levels. For the MES part, it is the refbox capability to generate orders, provide task goals to the robots, and collect, aggregate, and evaluate the overall performance. Whether the refbox also implements the SCADA level depends on the machines and communication access. For machines which provide direct access via Modbus, the refbox clearly performs this task. When machines provide a higher-level interface and possibly even combine information from separate parts or entities of a machine (e.g., through an embedded computer), this functionality is actually performed by the machines. The refbox system aggregates, organizes, and optimizes these capacities from the SCADA and MES level. In the first form, the refbox even directly influences the PLC and IO levels.

## 4 Knowledge-Based Instrumentation and Control

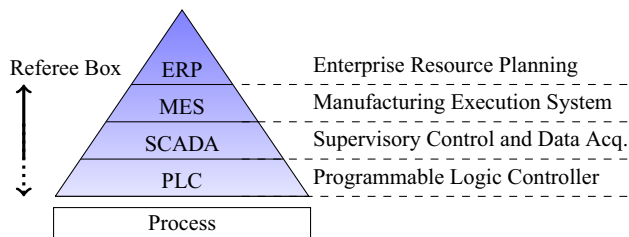
The presented refbox framework originated in the RoboCup Logistics League and was first used at the German Open 2013. It is based on a rule-based agent [20] developed previously for the competition. The framework uses the rule-based production system CLIPS (cf. [4], ch. 6). Rules encode specific situations in which the refbox must make a decision or trigger actions in the environment. Using such a system has several advantages, e.g., over encoding the rules as state machines or imperative programming paradigms. The rules provide a very clear specification of the relevant situations in the game and the used method has been optimized already for a high throughput of fact updates and involved triggering of rules (a typical RCLL game sees well beyond 1 M fact updates). Since all known information is represented explicitly, as are the rules, *instrumentation* is achieved by recording all relevant updates to the fact base and activated rules for later analysis. The refbox *controls* the game through sending messages to the robots and instructing the processing stations on the field.

The system also provides a robust and flexible core, which can then be extended for different scenarios, especially by an appropriate domain encoding. The RoCKIn project used this to adapt the refbox for their scenario. It is currently in preparation to be used in @Work. Currently, the systems exist as separate projects, with the intention to reunite them in the near future.

In the following, we briefly describe the CLIPS reasoning engine, before giving examples of the domain encoding in the different scenarios. We give a little more detailed information about the control and instrumentation aspects of the system and the communication infrastructure.

### 4.1 Rule-Based Reasoning Engine CLIPS

CLIPS [31] is a rule-based production system using forward chaining inference based on the Rete algorithm [9] consisting of three building blocks [10]: a fact base or working memory, the knowledge base, and an inference engine. *Facts* are basic forms representing pieces of information in the fact base. They usually adhere to structured types. The *knowledge base* comprises heuristic knowledge in the form of rules, and procedural knowledge in the form of functions. *Rules* are a core part of the production system. They are composed of an antecedent and consequent. The antecedent is a set of conditions, typically patterns which are a set of restrictions that determine which facts satisfy the condition. If all conditions are satisfied based on the existence, non-existence, or content of facts in the fact base the rule is activated and added to the agenda.



**Fig. 4** Enterprise-control system layers according to IEC 62264-1 [11]

The consequent is a series of actions which are executed for the currently selected rule on the agenda, for example to modify the fact base. *Functions* carry procedural knowledge and may have side effects. They can also be implemented in C++. In our framework, we use them to utilize the underlying robot software, for instance to communicate with the reactive behavior layer described below. CLIPS' *inference engine* combines working memory and knowledge base performing fact updates, rule activation, and agenda execution until stability is reached and no more rules are activated. Modifications of the fact base are evaluated if they activate (or deactivate) rules from the knowledge base. Activated rules are put onto the agenda. As there might be multiple active rules at a time, a conflict resolution strategy is required to decide which rule's actions to execute first. In our case, we order rules by their salience, a numeric value where higher value means higher priority. If rules with the same salience are active at a time, they are executed in the order of their activation.

## 4.2 Domain Encoding

The game and environment logics of the respective domains are encoded in distinct CLIPS rule sets. They process sensor information and incoming messages. Dynamic order or events, e.g., machine downtime, is created by time-based rules. The program shell provides the necessary interfaces to order action in the environment.

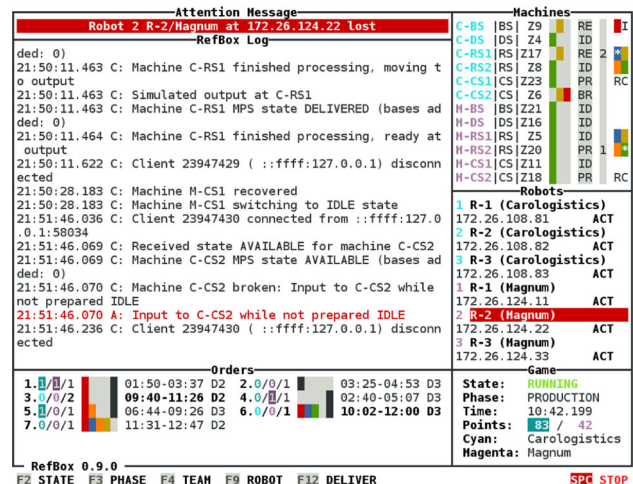
In the *RCLL*, rules encode specific situations and events when the refbox must act. The Modular Production System (MPS) programs are designed specifically that it performs only minimal action on an event (like retrieving a workpiece) and then wait for the refbox to order what has to happen next. This allows, for example, to have variable processing times. An example is shown in Fig. 5. During production (l. 4) it matches the situation when a ring station (RS) enters its processing state after a workpiece has been received (ll. 5–10) and that enough additional material (required bases) has been provided (ll. 11f.). It then updates the processed set and sets the machine lights (ll. 14f.) and orders the MPS station to mount the appropriate ring (ll. 16f.). Other rules (not shown) match certain

```

1 (defrule prod-proc-state-processing-rs
2   "Instruct RS to mount ring"
3   (declare (salience ?*PRIORITY_HIGH*))
4   (gamestate (state RUNNING) (phase PRODUCTION))
5   ?m <- (machine (name ?n) (mttype RS)
6         (state PROCESSING)
7         (proc-state ~PROCESSING)
8         (rs-ring-color ?ring-color)
9         (rs-ring-colors $?ring-colors)
10        (bases-added ?ba) (bases-used ?bu))
11   (ring-spec (color ?ring-color)
12             (req-bases ?rb&:(>= (- ?ba ?bu) ?rb)))
13 =>
14   (modify ?m (proc-state PROCESSING)
15             (desired-lights GREEN-ON YELLOW-ON))
16   (mps-rs-mount-ring (str-cat ?n)
17                     (member$ ?ring-color ?ring-colors))
18 )

```

**Fig. 5** CLIPS rule ordering an RS to mount a ring and setting the light signal (yellow and green steady on) once a workpiece has been provided (and thus the machine enters its processing state) (color figure online)



**Fig. 6** Control interface for the RCLL referee box

failure cases, like insufficient material. The 2015 rule set consists of about 150 rules and 40 fact structures. A typical game incurs about 1.5 million updates to the fact base (game time updates, received messages, machine instructions etc).

Figure 6 shows the control interface that combines all relevant information on the screen and allows to setup a game.

The RoCKIn task models are inspired by the RCLL. An overall task benchmark consists of the initial state and a coarse decomposition of the steps to be performed. The initial state, or inventory, describes the objects that the robot can find in specific locations or containers. Objects can either have unique identifiers (e.g. containers or trays to be inserted into machines) to allow tracking the production process or be multiple, generic objects packed in lots. The inventory can be modified dynamically, e.g. when a networked device has processed or delivered an item, but

also in a multi-robot scenario (which is not yet specified in the RoCKIn rules) robots could use this approach to share a world model. In order to represent the domain in the CFH, an object-oriented programming paradigm has been applied by using the CLIPS Object-Oriented Language (COOL) since it reflects the concepts of the domain and their links better than purely relying on functions.

By abstracting tasks into an inventory and orders a multitude of manufacturing and transportation scenario can be modeled. However, it is still of interest to identify further abstractions, for instance, to model a task like cleaning the factory floor, performing machine maintenance or ordering the robot to navigate to a specific goal under constraints like the orientation. Additionally, to support the robots it is worthwhile to investigate how to model machines generically, for instance, by specifying a machine's input/output relations or the machine's software interfaces which are operated by the robots. One representation of the input/output relations could be the production rules as already described in the RCLL.

### 4.3 Machine Control

In smart factories, actual manufacturing steps are provided by processing machines placed semi-statically in the environment. SCADA systems are necessary to process sensor data and control execution. Robots transport workpieces and handle these machines. Contrary to other architectures where devices directly communicate with each other, in our scenarios the refbox forms a central communication hub. Since a ubiquitous common network is required anyway, this is not a drawback. It also allows for access control, command verification, and execution monitoring of the performed steps. In the RCLL, processing stations are based on the Festo MPS. Currently, communication is based on Modbus (TCP/IP) over wifi to perform state machine synchronization. Since wifi communication is unreliable, a development effort has been started to use an embedded computer on the MPS for more robust communication using a more advanced network protocol in the future.

In @Work and RoCKIn the machines do not provide a common interface. While the conveyor belt is controlled via Modbus RTU, the quality control camera uses a USB interface and the drilling machine and force-fitting machine an Ethercat interface. Thus, a Raspberry Pi in front of each machine acts as a gateway to the CFH. Still, the individual commands vary with the specific machine. Therefore, the RCLL Refbox has been extended with a plugin mechanism. Each plugin takes care of the device-specific commands and provides a custom CLIPS interface to the CFH logic. From here, the interfaces can also be exposed to the robots.

### 4.4 Instrumentation and Benchmarking

An important aspect to use the presented scenarios as evaluation testbeds and for benchmarking is the acquisition of suitable data. The refbox core uses the document oriented database MongoDB for recording. It stores three principle types of data: all communication messages sent or received, each modification to the fact base and activated rules, and domain-specific information, e.g., game reports. In the RCLL, based on data of the 2014 world championship, and in-depth analysis has been performed [24]. Using the stored fact base updates, the games could be retraced in detail. A number of Key Performance Indicators from industrial scenarios have been proposed based on this analysis for further more detailed evaluation of the games.

For a logistics scenario with a focus on automated planning and scheduling, RCLL has demonstrated that tracing points via the machines in the testbed over time provides sufficient benchmarking data to evaluate a team's performance [24]. However, if robot manipulation and consequently also perception is required, more fine-grained benchmarking data is required. This holds especially, when a domain relies on further sub-domains, for instance, already a naïve subdivision of manipulation into planning and control. For each of these domains a variety of different criteria can be identified for benchmarking. Some criteria may require additional sensors in the testbed such as force sensors to evaluate a robot's physical interaction with the environment. RoCKIn relied on two main approaches to acquire robot data: Firstly, teams are requested to record the robot's internal data like raw sensor information such as laser scans or camera images, the robot's estimated pose or the pose estimation for an object w.r.t. a reference frame. Secondly, the robots are tracked by an external ground truth system which is based on a commercial motion capture system.

### 4.5 Robot Communication Infrastructure

Larger differences among the domains are evident in the infrastructure concerning communication between the refbox and robots, mostly due to the fact that the RCLL is a multi-robot scenario with two active teams at a time, while the others use a single robot exclusively on the field.

All domains use Google Protocol Buffer<sup>4</sup> (protobuf) for message encoding. It provides a rich definition language supporting hierarchical data structures, variable length lists, and required and optional fields. It has been integrated with CLIPS for reading and writing messages.

<sup>4</sup> <https://developers.google.com/protocol-buffers/>.

#### 4.5.1 Broadcast/Multicast Datagrams (UDP)

For robust communication, the refbox builds on experience from RoboCup soccer leagues (cf. Sect. 5). Wifi conditions are usually problematic during RoboCup competitions due to the large number of networks and devices connected. Protocols based on TCP often react poorly towards high package loss connections, especially for messages with timely updates. The TCP stack will try to re-transmit messages for too long, delaying later packages with already revised information. Furthermore, each time the connection is lost, a three-way handshake is required to re-open it. Doing this for multiple robots only adds to the problem. Therefore, a broadcast protocol based on UDP was chosen and information is periodically transmitted. Then, intermittently lost packages do not pose a problem as another (potentially already updated) message is sent shortly and no connection handshake is required. Broadcast has the disadvantage that it requires wifi networks to fall back to a low-bandwidth transmission mode. However, this can be alleviated switching to multicast communication with access points supporting multicast to unicast translation. The messages use a small framing protocol to identify the encapsulated protobuf message. A reference implementation using the Boost Asio library is available. Initially developed for the RCLL, theRoCKIn challenge and @Work league have adopted the same mode of communication.

#### 4.5.2 MQTT

In @Work, an additional experimental implementation of the communication based on message queuing telemetry transport [12] (MQTT) is being developed. It is a light weight stream-based messaging protocol. Therefore, it may incur the problems observed for TCP and described above (currently under investigation). It follows the publish-subscribe paradigm and requires a central message broker which is responsible for distributing messages to interested clients based on the topic of a message. The benefits are a more generic structure and naming of particular message channels. It also integrates with web technologies allowing to build, e.g., informative displays for visualization.

## 5 Related Work

RoboCup soccer leagues have used referee boxes for a long time—but which have so far never been autonomous. We describe and compare some relevant examples to our work. Furthermore, we provide some related work regarding benchmarking, as one of the most important features of our work.

## 5.1 Other RoboCup Leagues

In the *Middle Size League*<sup>5</sup> (MSL) the game is fully under human supervision and the refbox serves as communication interface between the human referees and the robot. The refbox has been in development for a long time seeing several implementations and communication protocols. The original version used a simple character-based protocol. Since 2009 a version written in Java used an XML-based protocol. Both of these used multicast communication for better robustness towards network failures. Since 2015 a new version is being developed using the Processing framework and stream-based (TCP) communication with a JSON [5] protocol. The refbox features no autonomy and is purely used as human-machine interface. It resembles a simple state machine with inputs being human operator inputs. It has basic logging of sent network messages and state changes.

In the *RoboCup Standard Platform League*<sup>6</sup> (SPL) a GameController (GC) application is used to instruct the gameplay [27]. It implements a state machine with six states controlling the overall game process and allows to penalize individual robots for misconduct. Several human referees supervise a game, one of which instructs the GC through a graphical interface. No autonomous decisions or automated event responses are implemented in this referee box. Pennisi et al. presented a system consisting of several RGBD cameras to acquire information about robots on the whole field [25], which could be used as the basis for introducing automated decision making in the GC. The GC collects some basic statistics, e.g., the goals scored and the misconducts penalized per team. Furthermore, it stores communication logs of messages sent and received. These data are less powerful in terms of evaluation capabilities compared to the RoboCup Industrial refbox. The SPL GC employs an UDP broadcast protocol with custom message encoding where the GC periodically sends out information to the robots.

The *RoboCup Small Size League*<sup>7</sup> (SSL) utilizes a referee box program which—similarly to the SPL—acts as an interface between human referees and the robots. A specialty of the SSL is an overhead camera system that tracks the robots and ball. This information is used by the teams for decision making and control, but may also be used for automatic refereeing. In order to foster this development, an “Autoref” Technical Challenge<sup>8</sup> has been declared for 2016. At this time, three proposals have been

<sup>5</sup> [http://wiki.robocup.org/wiki/Middle\\_Size\\_League](http://wiki.robocup.org/wiki/Middle_Size_League).

<sup>6</sup> <http://www.tzi.de/spl/>.

<sup>7</sup> [http://wiki.robocup.org/wiki/Small\\_Size\\_League](http://wiki.robocup.org/wiki/Small_Size_League).

<sup>8</sup> Technical Challenges in RoboCup are tasks in addition to the main task used to foster development of new capabilities in a league.



**Table 1** Comparison of robotic competition scenarios and their respective referee boxes

		Industry-inspired scenarios			Soccer leagues		
		Logistics	@Work	RoCKIn/ RockEU2	Middle Size League	Standard Platform League	Small Size League
Agency & Control	Functionality						
	1. Task management						
	a) Single task	✓	×	×	✓	✓	✓
	b) Choosing tasks from a predefined set	×	✓	✓ (FBM)	×	×	×
	c) Generation of task goals	✓	×	✓ (TBM)	×	×	×
	2. Environment agency						
	a) Exhibits agency	✓	(✓)	✓	×	×	×
	b) Production systems	12 MPS stations of 4 types	conveyor, rotating table	conveyor, drill, force- fitting			
	c) Automatically controlled	✓	(✓)	✓	×	×	×
Instrumentation & Benchmarking	3. Sensing						
	a) Arena sensors	MPS, (cam- era)				(Kinect [25])	cameras
	b) Machine feedback	state, work- piece ID	state	state, qual- ity			
	4. Benchmarking						
	a) External sensors	experimental		Vicon			
	b) Communication Logging	✓	✓	✓	✓	✓	✓
	c) State Logging	✓ (detailed)	✓ (detailed)	✓ (detailed)	✓	✓	✓
	5. Online run evaluation and scoring						
	a) Error detection (negative scores)	✓	×	×	×	×	(✓)
Visualization	b) Run evaluation (positive scores)	✓	×	partially (FBM)	✓	✓	✓
	c) Performance metrics	(✓)	×	offline	×	×	×
	6. Visualization						
	a) Team information	✓	×	×	✓	✓	✓
	b) Current Task (goals, destination)	✓	(✓)	✓	×	×	×
	c) Received points	✓	×	×	✓	✓	✓
	d) Map and robot positions	(✓)	(✓)	×	×	×	×
	e) External camera		arena				(✓)
	f) Machine state	(✓)	×	✓	×	×	×

submitted. The CMDragons team<sup>9</sup> use an event-based system implemented in C++ to trigger reactions to certain situations, the ER-Force team<sup>10</sup> uses a Lua-based imperative style situation detection, and the Tigers Mannheim team<sup>11</sup> implemented a state-based approach in Java. These efforts show that the implementation of an autonomous control system for robotic competitions is highly relevant. Due to the high throughput capabilities we would expect that the presented refbox could be used to implement an SSL autoref. The SSL uses a UDP multicast protocol using Protocol Buffer message encoding.

<sup>9</sup> <https://github.com/dzhu/ssl-autoref>.

<sup>10</sup> <https://github.com/robotics-erlangen/autoref>.

<sup>11</sup> <http://gitlab.tigers-mannheim.de/open-source/AutoReferee>.

Table 1 shows a tabulated comparison of the industrial scenarios with the described RoboCup soccer leagues. The comparison considers all information available to the best of the knowledge of the authors. We have also considered experimental or possible features like the SPL ground truth acquisition or the SSL Autoref challenge.

## 5.2 Benchmarking in a Smart Factory

Benchmarking plays an important role to evaluate and compare specific algorithms or integrated systems. However, identifying and defining benchmarks remains challenging, since it covers multiple dimensions. In the context of robotics and smart factories, such dimensions include, for instance

- the design of testbeds which reach from sensor and robot networks [3] to flight control systems [30]
- the design of benchmark for specific problem domains, like navigation and mapping [8, 29] or manipulation [16]
- the identification or definition of data sets [6, 14].

Benchmarking in an industrial testbed such as the RCLL, @Work, or RoCKIn adds another, orthogonal dimension, namely the availability of the data produced by the active testbed itself. This includes the sensors within the machines or sensors separately added to the factory, but also the data communicated between refbox, robots and machines. Due to the central role of the refbox, the data is already naturally available at this point. Logging is then merely a by-product of the refbox' knowledge-based design, because each rule activation is recorded. The holistic data acquired via this approach enables the instrumentation of the machines and robots within the smart factory and the in-depth analysis of the factory's overall performance.

## 6 Conclusion

The integration of an automated refbox is essential for robotic competitions in order to organize fair, transparent and reconstructible runs with a limited effort. As such, it supports the scientific and educational development of the league. Additionally, providing agency to an industrial environment with sensors that provide data and machines that must be instructed is crucial. Eventually we aim to be able to judge and evaluate games fully autonomously. However, this requires elaborate sensing capabilities that will take time to develop. Therefore, human referees will remain a part of the refereeing process for the foreseeable time.

In this paper we presented a common refbox framework for different competitions based on common requirements.<sup>12</sup> A generic architecture allows to re-use the same framework and core application in four presented scenarios, the RoboCup Industrial Logistics and @Work Leagues, and the RoCKIn and RockEU2 projects. The knowledge-based reasoning engine provides the necessary expressivity to model the scenarios rules and constraints, and implement the respective environment agency (control of machines). For this purpose it represents the competition rules in an abstract way. Furthermore, through automated logging of all communication messages, fact base updates, and activated rules a thorough analysis is possible (instrumentation of the environment). For applying the concept on different leagues mainly the knowledge base has to be replaced. A current development is to introduce plugins to extend the

capabilities of the reasoning engine, which will make it even simpler to replace, e.g., the communication infrastructure or integrate evaluation modules.

## References

1. Amigoni F, Bastianelli E, Berghofer J, Bonarini A, Fontana G, Hochgeschwender N, Iocchi L, Kraetzschmar G, Lima P, Matteucci M, Miraldo P, Nardi D, Schiaffonati V (2015) Competitions for benchmarking: task and functionality scoring complete performance assessment. *IEEE Robot Autom Mag* 22:53–61
2. Amigoni F, Bonarini A, Fontana G, Matteucci M, Schiaffonati V.: Benchmarking through competitions. In: European robotics forum—workshop on robot competitions: benchmarking, technology transfer, and education (2013)
3. Barbosa M, Bernardino A, Figueira D, Gaspar J, Goncalves N, Lima P, Moreno P, Pahlani, A, Santos-Victor J, Spaan M, Sequeira J.: Isrobotnet: a testbed for sensor and robot network systems. In: IEEE/RSJ international conference on intelligent robots and systems (IROS) (2009)
4. Brachman RJ, Levesque HJ.: Knowledge representation and reasoning. Elsevier, San Francisco (2004)
5. Bray T (2014) The JavaScript Object Notation (JSON) Data Interchange Format. RFC 7159, Internet Engineering Task Force
6. Calli B, Walsman A, Singh A, Srinivasa S, Abbeel P, Dollar AM (2015) Benchmarking in manipulation research. *IEEE Robot Autom Mag* 22:36–51
7. Dwiputra R, Berghofer J, Ahmad A, Awaad I, Amigoni F, Bischoff R, Bonarini A, Fontana G, Hegger F, Hochgeschwender N, Iocchi L, Kraetzschmar G, Lima PU, Matteucci M, Nardi D, Schiaffonati V, Schneider S (2014) The RoCKIn@Work Challenge. In: 45th international symposium on robotics (ISR)
8. Fontana G., Matteucci M, Sorrenti DG Rawseeds: building a benchmarking toolkit for autonomous robotics. In: Amigoni F, Schiaffonati V (eds) *Methods and experimental techniques in computer engineering*, SpringerBriefs in applied sciences and technology. Springer International Publishing, pp 55–68 (2014)
9. Forgy CL (1982) Rete: a fast algorithm for the many pattern/many object pattern match problem. *Artif Intell* 19(1)
10. Giarratano JC (2007) CLIPS reference manuals. <http://clipsrules.sf.net/OnlineDocs.html>
11. International Electrotechnical Commission (2013) Enterprise-control system integration—Part 1: models and terminology
12. International Electrotechnical Commission (2015) International Organization for Standardization: ISO/IEC DIS 20922: Information technology—message queuing telemetry transport (MQTT) v3.1.1
13. Kagermann H, Wahlster W, Helbig J (2013) Recommendations for implementing the strategic initiative INDUSTRIE 4.0. Final Report, Platform Industrie 4.0
14. Kasper A, Xue Z, Dillmann R (2012) The kit object models database: an object model database for object recognition, localization and manipulation in service robotics. *Int J Robot Res* 31:927–934
15. Kitano H, Asada M, Kuniyoshi Y, Noda I, Osawa E Robocup: the Robot World Cup Initiative. In: 1st Int. conference on autonomous agents (1997)
16. Moll M, Sucan IA, Kavraki LE (2015) Benchmarking motion planning algorithms. *IEEE Robot Autom Mag* 22:96–102
17. Niemueller T, Ewert D, Reuter S, Ferrein A (2013) Carologistics RoboCup team: autonomous referee box and visualization tool for the logistics league sponsored by Festo. RoboCup Grant

<sup>12</sup> The source of the RoboCup Industrial Referee Box is available at <https://github.com/robocup-industrial/rci-refbox>.

- Report Poster, RWTH Aachen University and FH Aachen UoAS. <https://www.carologistics.org/publications/2013/RC-Grant-2013/>
18. Niemueller T, Ewert D, Reuter S, Ferrein A, Jeschke S, Lakemeyer G (2013) RoboCup Logistics League Sponsored by Festo: a competitive factory automation testbed. In: RoboCup Symposium
  19. Niemueller T, Karpas E, Vaquero T, Timmons E (2016) Planning competition for logistics robots in simulation. In: WS on planning and robotics (PlanRob) at Int. Conf. on Aut. planning and scheduling (ICAPS)
  20. Niemueller T, Lakemeyer G, Ferrein A (2013) Incremental task-level reasoning in a competitive factory automation scenario. In: AAAI spring symposium 2013—designing intelligent robots: reintegrating AI
  21. Niemueller T, Lakemeyer G, Ferrein A (2015) The RoboCup logistics league as a benchmark for planning in robotics. In: WS on planning and robotics (PlanRob) at Int. Conf. on Aut. planning and scheduling (ICAPS)
  22. Niemueller T, Lakemeyer G, Ferrein A, Reuter S, Ewert D, Jeschke S, Pensky D, Karras U () Proposal for advancements to the LLSF in 2014 and beyond. In: ICAR—1st workshop on developments in robocup leagues (2013)
  23. Niemueller T, Lakemeyer G, Reuter S, Jeschke S, Ferrein A (2017) Cyber-physical systems—foundations, principles, and applications, chap. Benchmarking of Cyber-Physical Systems in Industrial Robotics—The RoboCup Logistics League as a CPS Benchmark Blueprint. Elsevier (**to appear**)
  24. Niemueller T, Reuter S, Ferrein A, Jeschke S, Lakemeyer G (2015) Evaluation of the RoboCup logistics league and derived criteria for future competitions. In: RoboCup Symposium
  25. Pennisi A, Bloisi DD, Iocchi L, Nardi D (2013) Ground truth acquisition of humanoid soccer robot behaviour. In: RoboCup symposium
  26. Reinhardt G, Krug S, Hüttner S, Mari Z, Riedelbauch F, Schlögel M (2010) Automatic configuration (plug&produce) of industrial ethernet networks. In: 9th IEEE/IAS international conference on Industry applications (INDUSCON), 2010, pp 1–6. doi:10.1109/INDUSCON.2010.5739892
  27. RoboCup SPL Technical Committee (2015) RoboCup SPL Technical Committee: RoboCup Standard Platform League (NAO) Rule Book 2015
  28. Schneider S, Hegger F, Hochgeschwender N, Dwiputra R, Moriarty A, Berghofer J, Kraetzschmar G (2015) Design and development of a benchmarking testbed for the factory of the future. In: IEEE International conference on emerging technologies and factory automation (ETFA)
  29. Sturm J, Engelhard N, Endres F, Burgard W, Cremers D (2012) A benchmark for the evaluation of RGB-D slam systems. In: IEEE/RSJ international conference on intelligent robots and systems (IROS)
  30. Wang Z, Gu F, He Y, Han J, Wang Y () Design and implementation of multiple-rotorcraft-flying-robot testbed. In: IEEE international conference on robotics and biomimetics (ROBIO) (2011)
  31. Wygant RM (1989) CLIPS: a powerful development and delivery expert system tool. *Comput Ind Eng* 17(1–4):546–549
  32. Zwilling F, Niemueller T, Lakemeyer G (2014) Simulation for the RoboCup logistics league with real-world environment agency and multi-level abstraction. In: RoboCup Symposium