Model-based Engineering and Analysis of Space-aware Systems Communicating via IEEE 802.11

Fenglin Han*, Jan Olaf Blech**, Peter Herrmann* and Heinz Schmidt**
*Norwegian University of Science and Technology, Trondheim, Norway
Email: {sih|herrmann}@item.ntnu.no
**RMIT University, Melbourne, Australia
Email: {janolaf.blech|heinz.schmidt}@rmit.edu.au

***Abstract*—We propose a model-driven development approach for autonomous control systems with emphasis on the physical space and the communication via wireless connections. In particular, we combine model-based engineering with simulation and emulation techniques for mobile communication. The design and implementation is done using our Reactive Blocks Framework. For the mobile communication we use the popular IEEE 802.11 WLAN protocol which is simulated using software tools in order to get estimations of connection delays. The spatial constraints are verified with our BeSpaceD tool. As an example, we present the design and verification of autonomous robots performing services in a large factory hall and coordinating by means of wireless communication which is based on several access points.**

I. Introduction

Recent technological development leads to a growing number of systems in which various mobile components act autonomously in a shared physical space. Examples for such systems abound in domains such as robotics, aeronautics, and automotive manufacturing. The autonomous systems must not interfere with each other in an undesired way which could result in accidents. Further, in many applications they form collaborative groups in order to perform joint tasks (e.g., several robots transport a heavy workpiece together on a construction site or coordinate their actions for warehouse automation). To achieve such a collaborative behavior, the components need to continuously interact with each other via wireless connections and the communication has to keep hard real-time limits. When developing such a system, one therefore has to consider communication-related issues (e.g., bandwidth limits occurring when many components are close together such that they are handled by a single access point). As shown in [1], the performance of a teleoperated robot system is strongly influenced by the quality of the communication environment.

Many industrial methods, e.g., Matlab/Simulink- and IEC 61131-based approaches, seek to build control systems solely from a model-based software engineering point of view. Others, e.g., OPNET [2], focus only on the communication network design and analysis.

In this paper, we propose bridging model-based development and network design. To this end, we study and evaluate a method for real-time control applications of wireless interconnected devices that takes both wireless network response times and spatiotemporal properties into account. In particular, we combine network simulation using Jemula [3] with spatial constraint solving using BeSpaceD [4], [5]. In extension of this previous work,

1) we apply constraint solutions to provide parameters for network simulations;
2) the results of the network simulations including failure probabilities and other statistical properties are then used to input further constraints into the constraint solver. For instance, if an access point fails, the neighboring ones may take over parts of its traffic, but at the expense of heavier loads.
3) The expected communication delays can be simulated with Jemula when one or more access points fail.

We also continue to use our model-based software engineering method Reactive Blocks[1] that allows us to specify system models by composing reusable sub-models, so-called building blocks [6], [7]. This approach enables us to create controllers for mobile components using the combination of these three methods. Our contribution is the combination of these methods as well as the exchange of models and the flow of analysis results.

The interaction of the three methods and tools is depicted in Fig. 1. The top section refers to the modeling of autonomous systems with Reactive Blocks. Moreover, as depicted in the center section, we simulate the network usage of the system to get significant predictions of the expected communication delay. In the current version of the toolchain, we restrict ourselves to the popular IEEE 802.11 series of WLAN protocols as interconnection technology and use the open source IEEE 802.11 simulator/emulator Jemula [3] to simulate WLAN usage. In the third step shown in the bottom section, the models and simulation data are used for verification. For instance, we can model check functional [7] and performance [8], [9] issues. This paper introduces another analysis technique, i.e., the use of BeSpaceD to prove the impact of communication delays for spatial properties of autonomous systems. Here, the BeSpaceD proofs take the knowledge about worst-case or average communication delays learned in the Jemula simulations into account.

[1]Until recently Reactive Blocks was named Arctis.

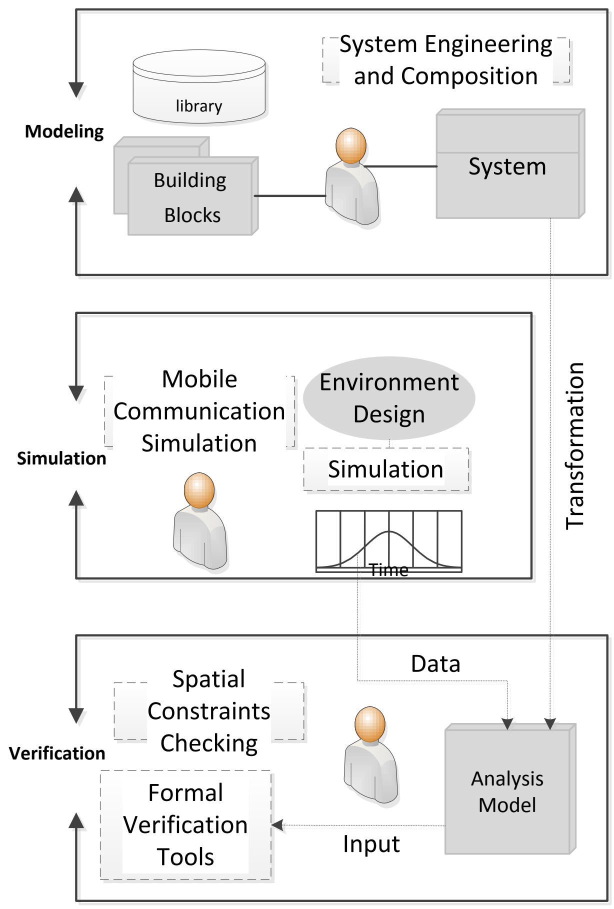

Fig. 1. Summary of the Approach.

The combination of these tools allows us to make useful predictions about the communication infrastructure of an embedded system already on the modeling level. If spatial properties are proven wrong, it is usually much cheaper to adapt the system infrastructure and the functionality of the mobile components on the modeling level than later when the system has already been implemented.

In Sect. II, we introduce the background of our approach. To facilitate understanding, we thereafter describe a mobile robot system-based scenario in Sect. III. The detailed control module description is discussed in Sect. IV followed by the definition of spatial properties in Sect. V. The WLAN communication simulation results for the scenario are introduced in Sect. VI. In Sect. VII, we explain how the simulation can be combined with the spatial analysis. The text is completed by a discussion about related work and a conclusion.

II. Background

The technology used in this work consists of the three tools Reactive Blocks, BeSpaceD and Jemula mentioned above. Further, relevant characteristics of the IEEE 802.11 protocol are discussed.

A. Reactive Blocks

The model-driven engineering method Reactive Blocks is a tool-supported approach for developing reactive concurrent software systems [7]. The systems are composed from models of reusable software components, so-called *building blocks*. The main ingredients of a building block are a UML 2.x activity diagram and an abstract *External State Machine* (ESM) [10]. Similar to a Petri net, the activity diagram models the detailed implementation logic as a token flow. To allow formal analysis, we supplemented the activity diagrams with a formal semantics allowing to order the token flows into reactive run-to-completion steps [11]. The ESM shows the abbreviated interface behavior of the building block as an abstract UML 2.x state machine. The concept enables us to specify recurrent sub-functionality by separate building blocks that can be specified once, stored in model libraries and reused in a drag-and-drop fashion in models. System models can be automatically analyzed for functional errors by a built-in model checker [7]. Further, executable Java code can be automatically generated [12].

An extension of the tool supports also the analysis of probabilistic real-time performance and safety issues. In particular, the time and probabilism properties can be formalized and analyzed using probabilistic timed automata-based formal verification [8], [13]. The formal analysis and verification is based on the two verification tools UPPAAL [14] and PRISM [15]. We established two extensions of the ESMs called *Real-Time External State Machine* (RTESM) [8], [16] and *Probabilistic Real-Time External State Machine* (PRTESM) [13] to model time-constrained behavior. This allows us to check hard real-time properties, e.g., proving that a system leaves a certain state within a period of time in order to carry out some safety preserving actions.

B. BeSpaceD

In [4], [5], we introduce BeSpaceD as a tool framework for specifying behavior of distributed systems and formally reasoning about them. BeSpaceD emphasizes on spatial behavior but is not restricted to this. It allows the verification of safety properties such as the absence of physical collisions between interacting robots and obstacles, the coverage of sensor ranges, or WLAN ranges. Specification is done using abstract datatypes out of a development environment supporting the Scala programming language. The abstract datatypes can be generated by Scala programs or by instantiation of other software. Checking and reasoning in BeSpaceD is realized using library functions creating verification goals. Verification goals are solved by standard tools such as SAT and SMT solvers or by specialized algorithms. For the purpose of this paper, specifications of spatial problems are done using the BeSpaceD constructs provided in a library inside code written in the Scala programming language.

C. IEEE 802.11 WLAN Delay Analysis

The IEEE 802.11 wireless local area network (WLAN) protocol is widely used for enterprise, home and public

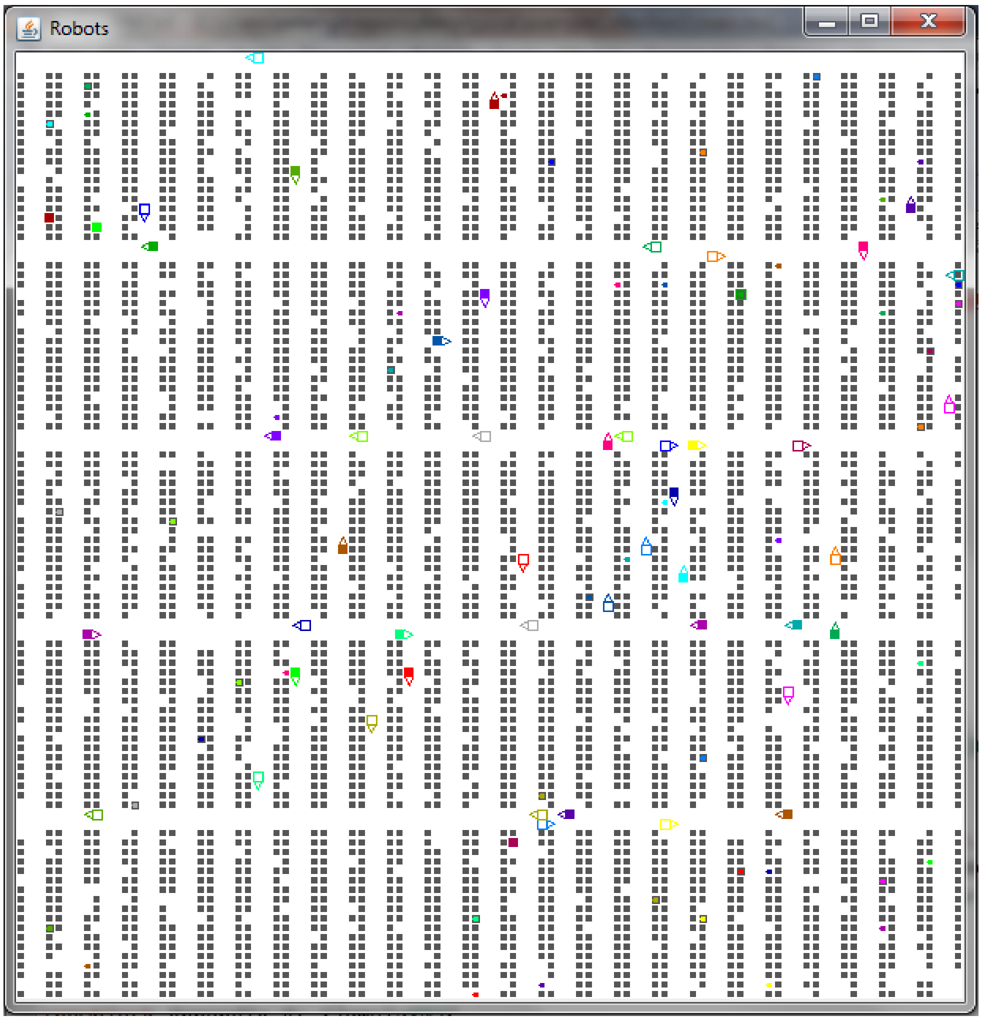

Fig. 2. Animation of the Warehouse Robot System with 50 Robots.

access networks. It consists of several protocol variations that define specifications for Media Access Control (MAC) and Physical (PHY) layer specifications. In the MAC layer, there are mainly two coordination functions, the *Distributed Coordination Function* (DCF) and the *Point Coordination Function* (PCF) that specify the media access and collision control. DCF is a distributed medium access scheme based on Carrier Sense Multiple Access using a Collision Avoidance (CSMA/CA) scheme with binary slotted exponential backoff. In the basic IEEE 802.11 MAC protocol using DCF, a station determines individually when to access the media. The station service responsible for information exchange is referred to as MAC Service Data Unit delivery (MSDU delivery). In contrast, PCF is a centralized media access function, in which the access point grants access to the different stations by polling.

Due to the competitive media access, DCF has the greatest impact on the performance of IEEE 802.11 wireless protocols. In our simulation we use the so called *Request/Clear To Send* (RTS/CTS) access mode which is the general mechanism to reserve the wireless communication channel in DCF. The WLAN latency is mainly composed of two parts, the transmission delay and the backoff delay. The transmission delay refers to the latency of transferring packets over the net and depends on the packet size (number of bits) and the transmission rate. The backoff delay refers to the lag of time a station needs to access the WLAN. The IEEE 802.11 MAC DCF uses the so-called binary slotted exponential backoff [17] to estimate the backoff delay. That is an analytical model computing the saturation throughput performance of DCF as a *stochastic* process. The model is easy to understand but can be hardly analyzed by machines due to the state space explosion problem. Therefore, we use emulators to simulate the communication collision and delay of stations to discover how the number of mobile stations can affect the delays and, in consequence, the overall network communication.

We apply Jemula802 and its kernel Jemula [3], which are open-source Java-based emulation tools to model IEEE 802-based communication. The Jemula emulation software has an event-based architecture and maintains an XML interface for the configuration of networks and systems. Like Reactive

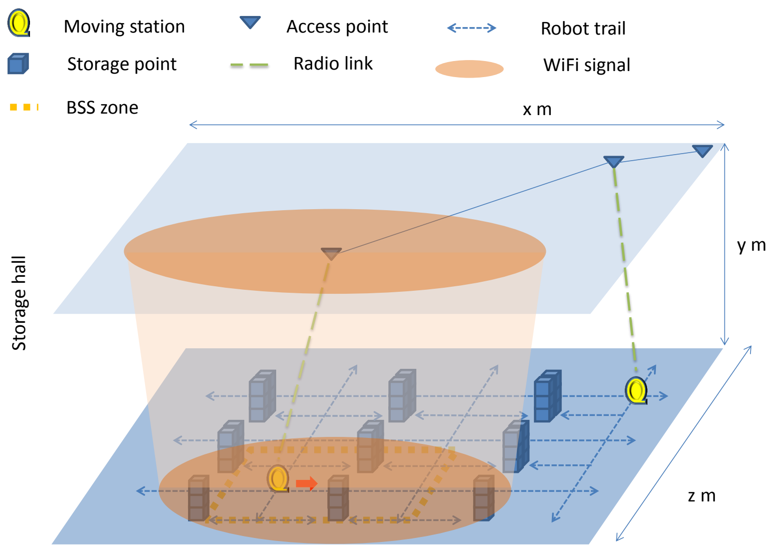

Fig. 3. Communication Infrastructure in the Storage Hall.

Blocks and BeSpaceD, Jemula is developed under the Eclipse framework.

III. Motivating Scenario

In this section, we introduce a scenario comprising a mobile distributed robot system followed by a closer discussion of its communication infrastructure.

A. *Warehouse Robot System*

We created a simulator of a mobile robot fulfilment system that contains multiple robots and handles goods in a warehouse. Our scenario is inspired by the Kiva robot system[2]. As depicted in Fig. 2, the robots fetch pallets (black rectangles) and transport them to other designated positions in the warehouse. Robots currently transporting a pallet are shown in the animation as filled colored rectangles while the unfilled ones refer to robots not carrying a pallet. The robots run on a line grid marking the floor. Lines are spaced one meter apart. Whenever a robot leaves a crossing, it signals its path to the next crossing to all the other robots in order to prevent that another robot heads towards the same crossing. When a new crossing is reached, another message to the other robots indicates that the crossing from which the robot left, is freed.

Due to the communication delay, nevertheless, up to four robots may be simultaneously on their way to the same crossing. As soon as a robot learns about such a conflict by receiving the respective signal from one of the other robots, it immediately stops and moves back to the crossing from which it was coming. Due to the physical layout of the system, the combined communication delay and reaction time leading to an emergency stop needs to take place within a second to avoid collisions. We showed in [8], [16] how maximum reaction times can be computed based on the system model. For our scenario, we found out that a reaction time of 200 ms is sufficient to stop the robot such that the maximum communication delay is 800 ms. The proof under which circumstances this maximum delay can be guaranteed by the system, is discussed later in this paper.

[2]https://www.youtube.com/watch?v=lWsMdN7HMuA.

B. *WLAN topology and environment*

We use a WLAN for the coordination and scheduling of the moving robots that is sketched in Fig. 3. The working space of the robots is modeled as a plane of width x, height y, and depth z. We assume that several access points are installed in the ceiling. The scenario is simplified by using the following assumptions:

1) We assume that neither the pallets and their content infer with the radio signals nor that there are obstacles between the access points and the robots. Thus, we suppose idealized conditions for the quality of the radio signals.
2) The access point antennas in the warehouse are arranged in a way that every robot is always in the range of one of them. To achieve that, we partition the warehouse into access areas and each access area is completely covered by the circular range of an access point. This will be discussed in detail in Sect. V.
3) Since robots can only physically interfere with each other if they are in close proximity, they will be either covered by the same access point or are at the border between two access points. Thus, a robot only needs to consider other robots and objects covered by the same or by neighboring access points. We reflect that in our simulations introduced in Sect. VI.

IV. Modeling the Controller in Reactive Blocks

Since the robot scenario consists of several robots of the same sort, we define the robot controller as a multi-session building block [18] in Reactive Blocks. This allows us to

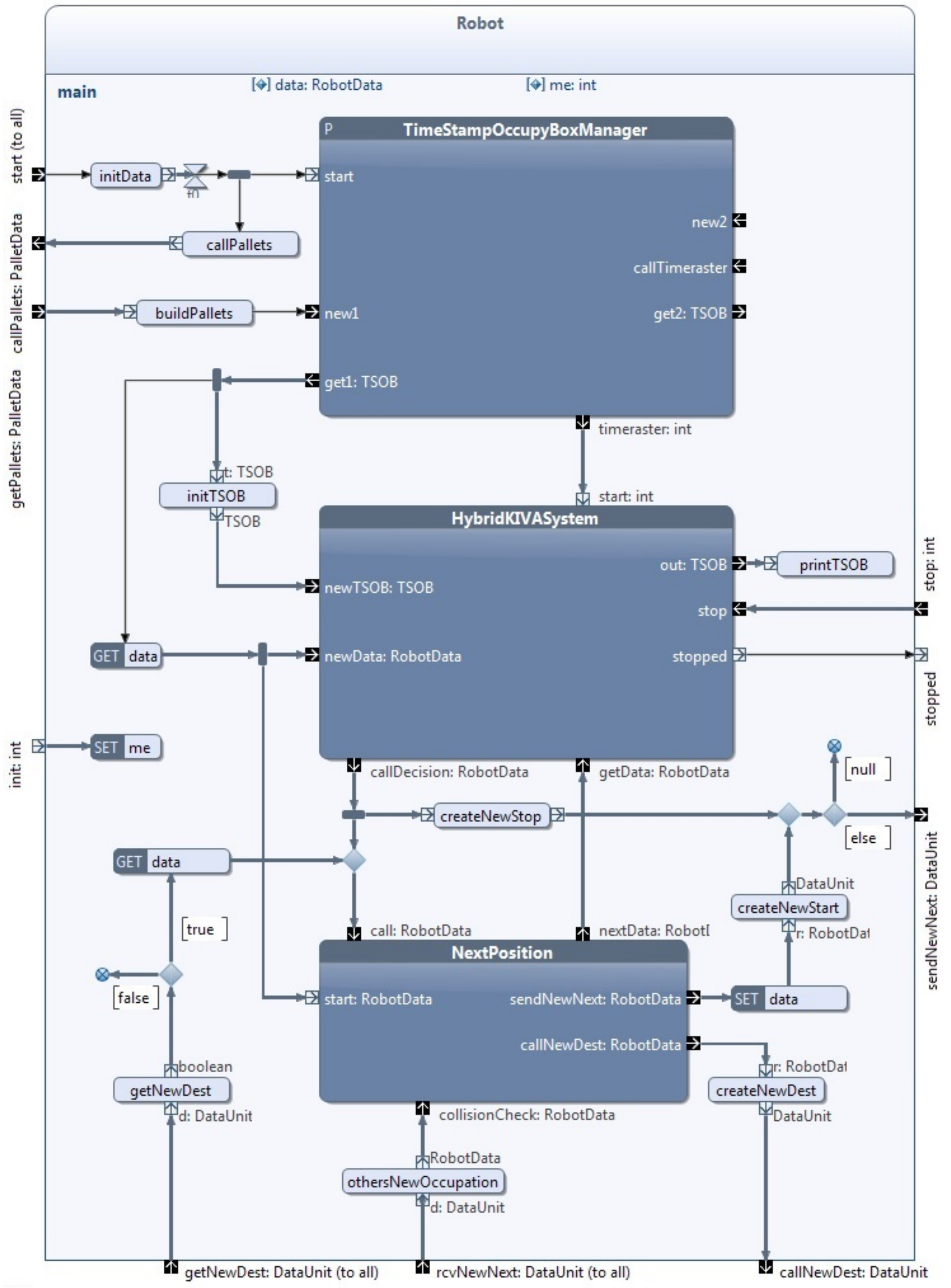

Fig. 4. Multi-session Building Block *Robot*.

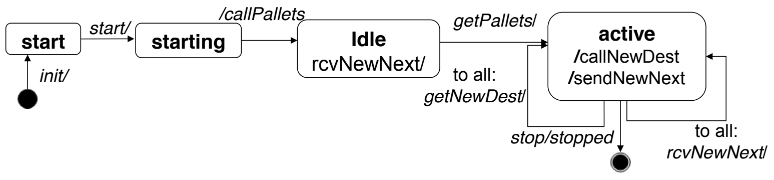

Fig. 5. ESM of Building Block *Robot*.

model an arbitrary number of control entities that can interact with each other as well as with their common environment. Figure 4 depicts the UML activity of the multiple-session building block that we named *Robot* while its ESM is found in Fig. 5. Each robot has a unique identifier stored in the variable *me*. The other variable *data* stores information such as the current position and direction of the robot, the pathes of other robots, and the positions of the pallets in the warehouse.

Robot contains three internal building blocks that each have their own UML activities and ESMs. *TimeStampOccupyBox-Manager* is taken from a library and manages the handling of data types carrying information about spatial properties and time of a robot. The building block *HybridKIVASystem* contains the basic logic of the robot control. Whenever the robot either reaches its destination or a crossing from which it can continue in various directions, a token is triggered passing pin *callDecision* of block *HybridKIVASystem*. This token is duplicated at the fork behind this pin and one copy forwards to operation *newStop* in which a Java method is executed that creates a message to inform the other robots that the crossing, from which the current one is reached, has been vacated. The token is forwarded to the parameter node *sendNewNext* enabling the environment of building block *Robot* to trigger a WLAN message to all other robots. The other token copy is forwarded to the building block *NextPosition* which contains the routing algorithm of the robot.

If the robot did not yet reach its destination, in *NextPosition* the new path is computed and forwarded via pin *nextData* to block *HybridKIVASystem*. In parallel, a flow leaves pin *sendNewNext* and forwards to a set-method for the local variable *data* that contains relevant data of the robot and its environment. Thereafter, a data unit is generated to inform the other robots of the next crossing to which the robot proceeds. The data unit is forwarded to parameter node *sendNewNext* in order to instantiate the according WLAN communication. If the robot reached its destination, a flow is issued passing via parameter node *callNewDest* to the system control which might assign a new task to the robot. The answer from the system control is received via a flow through *getNewDest*. As indicated by the filter *to all*, the new destination is sent to all robots since each one stores which pallets are currently transported by a robot.

Messages indicating the current positions and paths of other robots are received via parameter node *rcvNewNext* from where they proceed to operation *othersNewOccupation*. In the corresponding Java method, the received data is stored in variable *data*. The message is forwarded to building block *NextPosition* to be checked if the other robot is on a collision course with the local one. In this case, a token is sent via pin *nextData* to block *HybridKIVASystem* that causes the emergency stop of the robot as well as the retreat to the crossing from where the robot left.

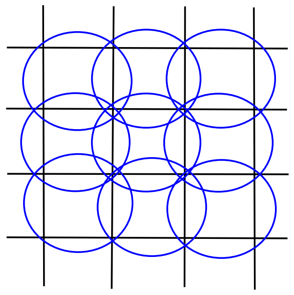

Fig. 6. Spatial Arrangement of Access Points.

V. Communication Access Ranges

The warehouse layout discussed in Sect. III-B raises a number of spatial issues that have to be addressed by the overall system design:

1) The access points have a limited range and we must ensure that every point in the storage hall in which robots may act, is sufficiently covered by access points. That holds particularly, when unlike to our assumptions in Sect. III-B, the storage hall contains obstacles which may distort radio signals.
2) An access point may fail with a certain probability. In this case, some robots in the coverage area of the broken access point may be covered by the neighboring access points while other robots may experience communication loss. In the former case, the access points taking over may get saturated. We verify properties of saturation, spatial aspects and failure probability to foster risk analysis. To guarantee high availability of continuous communication access for all robots, we require this failure probability to be below a minimal threshold.
3) Robots are moving through the hall and it may be that several of them are in proximity to each other. That does not only raise the probability of accidents between robots but might also increase the communication delay of the access points in this zone. We have to find out if there is sufficient bandwidth to guarantee certain maximum communication delays even if many robots are close together.

BeSpaceD [4], [5] allows us to establish spatial models for access point ranges including probabilities for, e.g., failures. For example, we can define a higher likelihood of a communication failure if a robot is farer away of the nearest access point. We assume that the robots keep a perfect, error-free connection with an access point if they are within a certain limited radius but that the QoS deteriorates when the robot is outside this radius. Furthermore, we can formalize behavioral properties of robots like positions and paths and also situation-dependent constraints on the channels [9], [13]. Ranges of access points tend to have a circular shape. But we can also partition the ranges in other shapes like the rectangular partitioning that is shown in Fig. 6. Here, a square describes an area fully covered by one access point. Particularly the edges of a rectangle may also be covered by other access points.

For our robot scenario, we used the BeSpaceD tool to check the above mentioned consistency and safety properties regarding the coverage. In particular, we checked consistency properties, e.g., is there a sufficient coverage for a given number of robots and safety properties: Given a probability distribution for communication delays, can we guarantee an upper bound for communication time between an access point and a robot? This is closer discussed in Sect. VII.

Below, we list a small code fragment written in Scala for the specification of a simplified coverage problem consisting of circular access point ranges for a rectangular factory hall at a given point of time below:

```
   def coverage =
     IMPLIES (AND(Owner ("WLAN_RANGE"),
     Prob(1.0)), BIGAND(
                OccupyCircle(15,15,40)::
                OccupyCircle(35,15,43)::
                ...
                OccupyCircle(65,130,42)::
                Nil ));

     def factoryhallarea =
      IMPLIES (AND(Owner ("FactoryHall"),
      Event("Operation")),
         OccupyBox(10,10,100,150)
     );
```

The example uses implications for specification. The idea is that time, event, probability, and ownership combinations imply space occupation, i.e., space occupation is provided for a particular aspect (here, the range of an access point `WLAN_RANGE`) and the probability that communication is successful. For mobile components the space occupation may vary depending on the time. Thus, by using spatiotemporal models for both the access point ranges and for robots, one can for instance prove with BeSpaceD whether a robot is always covered by circles that have probability 1.0. The space occupation part in the example uses a *logical conjunction* to combine different geometric shapes. In the example we use circles `OccupyCircle` with different coordinates and radius.

BeSpaceD based checking is done by calling library functions. An example for an spatial inclusion test is provided below. It makes use of breaking specifications (here, a box and circles) down into comparable entities `unfoldInvariant` and carrying out the actual inclusion test `inclusionTestsBig`:

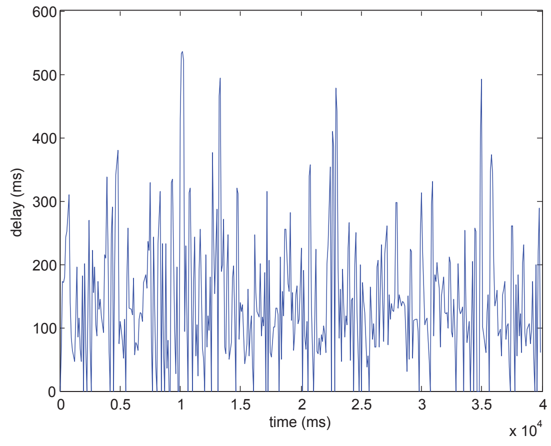

Fig. 7. MSDU Delivery Delay Simulation.

```
inclusionTestsBig(
   unfoldInvariant(
      OccupyBox(10,10,200,250))::Nil,
   unfoldInvariant(
      BIGAND(
         OccupyCircle(15,15,10)::
               ...
         OccupyCircle(35,16,12)::
         Nil ))::Nil))
```

VI. Simulation Results

The second and third spatial properties mentioned above can only be proven if the WLAN can guarantee a certain maximum communication delay. We use Jemula802 [3] to simulate the IEEE 802.11a physical layer (PHY) at the wave band of 5 GHz that allows up to a transmission rate of 54 Mb/s. Each mobile station, i.e., each robot, uses an omnidirectional antenna to send and receive the control and data packets. We assume that the maximum packet size of the control data is 200 bytes which corresponds to the fact that the identifier of the robot as well as information about the reached and vacated crossings are transported. Jemula802 emits packets at time intervals following a negative exponential distribution. For our example, the tool generates a mean load of 0.9 Mb/s. The packet size varies by a small amount which is uniformly distributed.

We configured a traffic generator for each mobile robot under the same access point. The generator emits network packets in the form of MAC Service Data Unit (MSDU) deliveries [19] and the simulated transmission time corresponds to the network communication latency for the robots. We plotted the communication delays simulated for 40 s and simulated several scenarios with different numbers of robots. Figure 7 shows the result of this simulation when an access point interconnects 50 mobile robots. Figure 8 depicts the delay distribution histogram of the simulation results. The simulation generates 400 MSDU packets, and we used the *matlab* tool to get several useful indices: The maximum delay is 535.7 ms and the mean or expected value of the communication delay is 142.9 ms. Compared with the maximum delay of 281.3 ms, and the expected value 76.7 ms when we consider

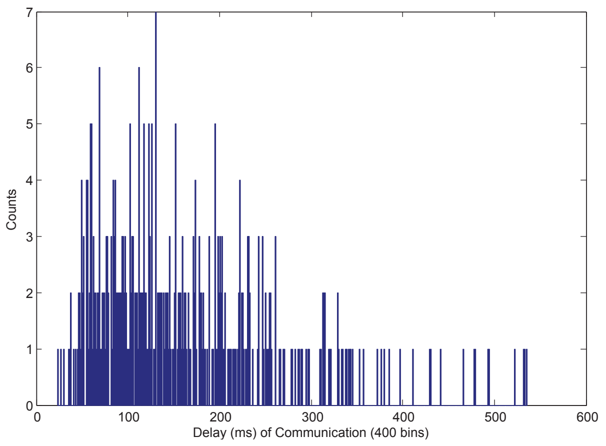

Fig. 8. Communication Delay Distribution.

only 20 robots communicating via the same access point, the delay shows a significant growth. Thus, while the maximum communication delay is significant, it is sufficiently below the maximum acceptable delay time of 800 ms necessary to avoid an accident between robots. The same result was also found in the other simulations with different parameters, e.g., numbers of robots or size of access point ranges. Since the throughput of a wireless channel is affected by multiple factors, e.g, size of delivered MSDUs, PHY layer modes used, and bandwidth offered to the communication channel, the mobile stations are adjustable according to need.

VII. COMBINING THE RESULTS

Our underlying method combines spatial constraint specification and solving with network topology and packet traffic simulation. On the one hand, we use spatial analysis results to parameterize the network simulation. On the other hand, we take simulation results including failure probabilities into a further spatial analysis.

BeSpaceD analyzes the satisfaction of spatial constraints listed in Sec. V, such as sufficient coverage of the warehouse by access points so that all robots are always in range. These analysis results are applied to prime the Jemula simulations. For example, we determine the topology of base stations including the number of channels available in a certain area. In turn, this determines the parameters for and the number of Jemula runs necessary to cover all relevant use cases.

For further constraint analysis, BeSpaceD takes the Jemula simulation results as input. For instance, if an access point fails, the neighboring ones may take over parts of its traffic, but at the expense of heavier loads. The expected communication delays can be simulated with Jemula when one or more access points fail. In BeSpaceD, we can now use these simulation results together with the probabilities that we assume for the failures of access points as well as spatial properties (e.g., the likelihood that a robot in the zone of a failed access point is at a place also covered by one of its neighbors) to verify if the overall probability that all robots can communicate with each other within a certain period of time, is greater or equal a certain value p. The value p (e.g., 99.999% that a data exchange is completed within 800 ms) helps to estimate the average risk of costs caused by malfunctions resulting in accidents. BeSpaceD was used to check several scenarios based on Jemula input.

Based on the risk analysis, one can decide if other protective mechanisms are necessary, for instance a function in the robot control logic that continuously checks if an active connection is available and immediately stops the robot if that is not the case.

For the third property of Sect. V, i.e., the check if there is sufficient bandwidth when several robots are in a certain area, the Jemula simulations are also used. Particularly, we simulate up to which number of robots the maximum communication delay is still bearable.

VIII. RELATED WORK

The surveys in [20], [21] show that due to the memory and time cost, simulation-based verification is popular in industry and software practice. It is often seen as simple and straightforward. In [22], however, Dill states that simulation-based verification does not cope with increasing system complexity since it is getting more and more difficult to select suitable test cases. Since formal verification is sometimes too complex to be automatically executed by machines and highly laborious when carried out manually, we seek an approach combining simulation and formal verification to achieve high quality software verification results in an acceptable period of time.

For the translation from state machine-based component models to Petri net analysis, [23] proposed a set of translation strategies from state machines using the history attribute to a class of non-autonomous Petri nets named Input-Output Place Transition Nets (IOPT nets).

There are plenty of network simulation tools available and used widely in research. Besides Jemula, simulation and emulation tools for IEEE 802.11 wireless networks also comprise ns2 [24], OPNET [2], and NCTUns [25]. Experimental research using these tools shows that the performance of a teleoperated robot system is strongly influenced by the quality of the communication environment [1]. Also in [26], the benefits of extensive use of wireless technologies in automation and robotics are discussed.

The IEEE 802.11 series of wireless protocols has several flavors in application to industrial robotics, among which IEEE 802.11a is considered as the most suitable existing solution. An extensive survey on wireless sensor network emulators and simulators is discussed in [27]. In [28], an experimental assessment of indoor propagation of WiFi signals with access points as transmitters operating at a frequency of 2.4 *GHz* is undertaken. The power density distribution of the access point antenna is achieved by movable laptop computers comprising wifihopper software. The measured and simulated results are compared to get a correlation coefficient factor (ρ), which indicates a good agreement between measurement and simulation results.

IX. CONCLUSION AND FUTURE WORK

In this paper, we studied and evaluated a method for real-time control applications of wireless interconnected devices.

Our method takes wireless network response times and spatiotemporal properties into account. We exemplified this by using mobile robots carrying out tasks in a warehouse. The ingredients of our method comprise the modeling of the system software and the simulation and analysis of the local access point networks integrated with spatial constraint solving.

In the next step, we seek further integration of the system development with the simulation and verification tools. In particular, we are aiming at directly encapsulating the simulation generator into a separate building block in our Reactive Blocks environment.

Acknowledgment

We express our gratitude to Professor Yuming Jiang at the Norwegian University of Science and Technology for his useful suggestions and discussions during the research work.

References

[1] Z. Szanto, L. Marton, P. Haller, and S. Gyorgy, "Performance Analysis of WLAN based Mobile Robot Teleoperation," in *IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*. IEEE Computer, 2013, pp. 299–305.

[2] X. Chang, "Network simulations with OPNET," in *31st Conference on Winter Simulation: Simulation — a Bridge to the Future (WSC'99)*, vol. 1. ACM, 1999, pp. 307–314.

[3] L. Berlemann and S. Mangold, *Cognitive Radio and Dynamic Spectrum Access*. John Wiley & Sons, 2009, ch. Appendix A: Jemula802.

[4] J. O. Blech and H. Schmidt, "BeSpaceD: Towards a Tool Framework and Methodology for the Specification and Verification of Spatial Behavior of Distributed Software Component Systems," arXiv.org, Tech. Rep., 2014.

[5] ——, "Towards Modeling and Checking the Spatial and Interaction Behavior of Widely Distributed Systems," in *Improving Systems and Software Engineering Conference*, 2013.

[6] F. A. Kraemer, "Engineering Reactive Systems: A Compositional and Model-Driven Method Based on Collaborative Building Blocks," Ph.D. dissertation, Norwegian University of Science and Technology, 2008.

[7] F. A. Kraemer, V. Slåtten, and P. Herrmann, "Tool Support for the Rapid Composition, Analysis and Implementation of Reactive Services," *Journal of Systems and Software*, vol. 82, no. 12, pp. 2068–2080, 2009.

[8] F. Han, P. Herrmann, and H. Le, "Modeling and Verifying Real-Time Properties of Reactive Systems," in *18th International Conference on Engineering of Complex Computer Systems (ICECCS)*. IEEE Computer, 2013, pp. 14–23.

[9] P. Herrmann, J. O. Blech, F. Han, and H. Schmidt, "A Model-based Toolchain to Verify Spatial Behavior of Cyber-Physical Systems," in *2014 Asia-Pacific Services Computing Conference (APSCC)*, 2014.

[10] F. A. Kraemer and P. Herrmann, "Automated Encapsulation of UML Activities for Incremental Development and Verification," in *Model Driven Engineering Languages and Systems (MoDELS)*, ser. LNCS 5795. Springer-Verlag, 2009, pp. 571–585.

[11] ——, "Reactive Semantics for Distributed UML Activities," in *Joint WG6.1 International Conference (FMOODS) and WG6.1 International Conference (FORTE)*, ser. LNCS 6117. Springer-Verlag, 2010, pp. 17–31.

[12] F. A. Kraemer, P. Herrmann, and R. Bræk, "Aligning UML 2.0 State Machines and Temporal Logic for the Efficient Execution of Services," in *8th International Symposium on Distributed Objects and Applications (DOA06)*, ser. LNCS 4276. Springer-Verlag, 2006, pp. 1614–1632.

[13] F. Han, J. O. Blech, P. Herrmann, and H. Schmidt, "Towards Verifying Safety Properties of Real-Time Probability Systems," in *11th International Workshop on Formal Engineering approaches to Software Components and Architectures (FESCA)*. EPTCS, 2014.

[14] J. Bengtsson, F. Larsson, P. Pettersson, W. Yi, P. Christensen, J. Jensen, P. Jensen, K. Larsen, and T. Sorensen, "UPPAAL: A Tool Suite for Validation and Verification of Real-Time Systems," in *Hybrid Systems III*, ser. LNCS 1066. Springer-Verlag, 1996, pp. 232–243.

[15] M. Kwiatkowska, G. Norman, and D. Parker, "PRISM 4.0: Verification of Probabilistic Real-Time Systems," in *23rd International Conference on Computer Aided Verification (CAV'11)*, ser. LNCS 6806. Springer-Verlag, 2011, pp. 585–591.

[16] F. Han and P. Herrmann, "Modeling real-time system performance with respect to scheduling analysis," in *6th IEEE International Conference on Ubi-Media Computing*. IEEE Computer, 2013, pp. 663–671.

[17] G. Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function," *IEEE Journal on Selected Areas in Communication*, vol. 18, no. 3, pp. 535–547, 2006.

[18] F. A. Kraemer, R. Bræk, and P. Herrmann, "Synthesizing Components with Sessions from Collaboration-Oriented Service Specifications," in *SDL-Forum*, ser. LNCS 4745. Springer-Verlag, 2007, pp. 166–185.

[19] S. Mangold, S. Choi, G. R. Hiertz, O. Klein, and B. Walke, "Analysis of IEEE 802.11e for QoS Support in Wireless LANs," *Wireless Communications*, vol. 10, no. 6, pp. 40–50, 2003.

[20] J. Jose and S. A. Basheer, "A Comparison of Assertion Based Formal Verification with Coverage driven Constrained Random Simulation, Experience on a Legacy IP," Wipro Technologies Reports, Tech. Rep., 2009.

[21] W. K. Lam, *Hardware Design Verification: Simulation and Formal Method-Based Approaches*. Prentice Hall, 2005.

[22] D. L. Dill, "What's between simulation and formal verification?" in *Design Automation Conference*, 1998, pp. 328–329.

[23] R. Pais, L. Gomes, and J. P. Barros, "From UML state machines to Petri nets: History attribute translation strategies," in *37th Annual Conference on IEEE Industrial Electronics Society (IECON 2011)*, 2011, pp. 3776–3781.

[24] T. Issaryakul and E. Hossain, *Introduction to Network Simulator NS2*, 2nd ed. Springer-Verlag, 2012.

[25] S. Y. Wang and C. C. Lin, "NCTUns 5.0: A Network Simulator for IEEE 802.11(p) and 1609 Wireless Vehicular Network Researches," in *68th IEEE Vehicular Technology Conference*. IEEE Computer, 2008, pp. 1–2.

[26] R. Calcagno, F. Rusina, F. Deregibus, A. S. Vincentelli, and A. Bonivento, "Application of Wireless Technologies in Automotive Production Systems," *VDI Berichte*, vol. 1956, pp. 57–58, 2006.

[27] M. Imran, A. M. Said, and H. Hasbullah, "A Survey of Simulators, Emulators and Testbeds for Wireless Sensor Networks," in *International Symposium in Information Technology (ITSim)*, vol. 2, 2010, pp. 897–902.

[28] B. M. Abaoy and K. M. Quboa, "Environmental Safety Standards and WLAN Indoor Propagation," in *20th Telecommunications Forum (TELFOR)*, 2012, pp. 13–16.