Real-time Human-in-the-loop Simulation with Mobile Agents, Chat Bots, and Crowd Sensing

Stefan Bosse
University of Koblenz-Landau
Faculty Computer Science,
Koblenz
Email: sbosse@uni-bremen.de

Raphael Heiberger
BIGSSS, University of Bremen,
Bremen
raphael.heiberger@uni-bremen.de

Abstract— **Modelling and simulation of social interaction and network formation is of high interest in multiple disciplines and fields of application ranging from fundamental social science to smart city management. Today, the ubiquitous usage of digital social media has relevant impact of social interaction. mobility, and opinion making. Virtual worlds, i.e., simulations and games, are commonly closed and rely on artificial social behaviour and synthetic sensory information generated by the simulator program or using data collected off-line by surveys. In this work, a crowd sensing and social data mining framework using mobile agents is introduced. The crowd sensing via chat bots using mobile agents creates augmented virtuality and reality by extending simulation worlds with real world interaction and vice-versa. The simulation world interacts with real world environments, humans, machines, and other virtual worlds in real-time. Mobile agents are closely related to bots that can interact with humans via chat dialogs. Among the mining of physical sensors (temperature, motion, position, light) of mobile devices like smart phones, mobile agents can perform crowd sensing by participating in question–answer dialogs via a chat blog (provided by smart phone Apps or integrated in WEB pages and social media). Additionally, mobile agents can act as virtual sensors (offering data exchanged with other agents). Virtual sensors are sensor aggregators performing sensor fusion in a spatially region. A simple social network analysis case-study based on an extended Sakoda social interaction model used for simulation with mobile crowd sensing demonstrate the capabilities of the new hybrid simulation method.**

I. INTRODUCTION

Today administration and management of public services and infrastructure relies more and more on user and ubiquitous data collected by many domestic and private devices including smart phones and Internet services. People use social digital media extensively and provide private data, enabling profiling and tracing. User data and user decision making has a large impact on public decision making processes, for example, plan-based traffic flow control. Furthermore, intelligent behaviour, i.e., cognitive, knowledge-based, adaptive, and self-organising behaviour based on learning, emerges rapidly in today's machines and environments. Social science itself exerts influence on public opinion and decision formation.

Traffic is a classic example for group decision making with (social) neighbouring interaction, but commonly traffic control relies on sensor processing only [1]. Car traffic control in smart cities can be achieved by global traffic sign synchronisation. But pedestrian, bicycle, and domestic traffic usage cannot controlled this way. A future vision to control domestic traffic flows is the deployment of digital and social medsia with chat bots to influence people decision making regarding goal driven mobility, sketched with the methods introduced in this paper.

Surveys play another important role for tmodelling and understanding of interaction patterns. Artificial Intelligence (AI) and chat bots shift classical survey methods towards computational methods introducing possible implications, i.e., usage of different data, different methods, exposition to different threats to data quality (concerning sampling, selection effects, measurement effects, data analysis), and different rules of inference are likely to result in comparably different conclusions, public reports, and hence in different input to public opinion formation. Among field studies, simulations can contribute to the investigation and understanding of such interactions.

Hardware and software robots can be considered close together in a generalised way by an association to the agent model. One prominent example for a software robot is a chat or social bot. Additionally, real and artificial humans can be represented by the agent model, too. Multi-agent systems enable modelling of agent interaction and emergence behaviour. Agent-based modelling and simulation is a suitable methodology to study interaction and mobility behaviour of large groups of relevant human actors, hardware, and software robots, using methods with inherent elements of AI (e.g. learning algorithms) or which use data which may be influenced partly by bots.

Agent-based methods are established for modelling and studying of complex dynamic systems and for implementing distributed intelligent systems, e.g., in traffic and transportation control [2] [1]. Therefore, agent-based methods can be divided into three main classes [3]:

1. Agent-based Modelling (ABM) - Modelling of complex dynamic systems by using the agent behaviour and interaction model ⇒ ***Physical agents***
2. Agent-based Computing (ABC) - Distributed and parallel computing using mobile agents related to mobile software processes ⇒ ***Computational agents***
3. Agent-based Simulation (ABS) - Simulation of agents or by using agents
4. Agent-based Modelling and Simulation (ABMS)

This work was not supported by any organization

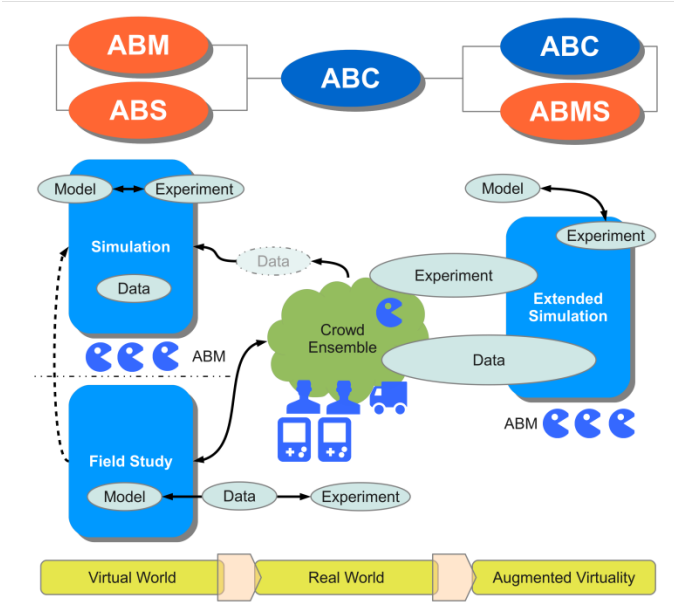

Fig. 1: (Top) The three different agent-based methods with overlapping relations - ABM: Agent-based Modelling, ABC: Agent-based Computing, ABS: Agent-based Simulation ABMS: A. Modelling and Simulation (Bottom) Transition from separated field studies and simulation towards extended simulation merging real and virtual worlds

Two promising fields of current studies in computer science are Data Mining (DM) and Agent Based Modeling and Simulation (ABMS) [3]. Agent-based simulation is suitable for modelling complex social systems with respect to interaction between individual entities, manipulation of the world, spatial movement, and emergence effects of groups of entities. The main advantage is the bottom-up modelling approach composing large-scale complex systems by simple entity models. The main disadvantage of ABM is the (over-) simplified entity behaviour and simplification of the world the entities acting in. Commonly, simulation bases on synthetic data or data retrieved by field studies. Many simulations and models lacking of diversity existing in real world. Commonly, sensor and model data (parameters) used in simulations (virtual world) is retrieved from experiments or field studies (real world), shown in Fig. 1. But there is neither a feedback from the virtual to the real world nor an interaction of the real world with the virtual world.

ABS and ABMS is commonly constructed using collections of condition-action rules to be able to percept and react to their situation, to pursue the goals they are given, and to interact with other agents, for example, by sending them messages [4]. The *NetLogo* simulator is an example of an established ABS tool used in social and natural sciences [5], but is limited to behavioural simulation only (ABM domain). In this work, a different simulation approach combining ABM, ABC, and ABS methodologies, is used deploying the widely used programming language JavaScript. JavaScript is used increasingly as a generic programming language for ABC and ABS [6] [7].

Mobile devices like smart phones are valuable sources for social data [8], either by participatory crowd sensing with explicit participation of users providing first class data (e.g., performing surveys or polls) or implicitly by opportunistic crowd sensing collecting secondary class data, i.e., traces of device sensor data delivering, e.g., actual position, ambient conditions, network connectivity, digital media interaction, and so on. Crowd sensing and Social Data Mining as a data source contribute more and more to investigations of digital traces in large-scale machine-human environments characterised by complex interactions and causalities between perception and action (decision making).

But mobile devices are not limited to being sensors. They can be used as actuators, too, by interacting with the user via the chat dialog or social media that can effect the user behaviour (e.g., mobility decisions) .

It is difficult to study such large-scale data collection, data mining, and their effect on societies, domestic services, and social interaction in field studies due to a lack of reliable data and complexity. Agent-based modelling of socio-technical systems is well established [9] and can be applied for smart city management, however commonly applied in an artificial world., i.e., a simulation is performed in virtual reality worlds only to derive and proof models under hard limitations. In this work, a new concept and framework for augmented virtual reality simulation is introduced, suitable, but not limited to, to investigate large-scale socio-technical systems. Mobile agents are used already successfully in-field crowd sensing [10]. In this work, mobile agents are used to combine in-field ubiquitous crowd sensing, e.g., performed by mobile devices, with simulation.

A chat bot agent is capable to perform dynamic dialogs with humans to get empirical data into the simulation and to propagate synthetic data from the simulation world into the real world. The chat bot can act as an avatar providing information for users, e.g., for optimised and dynamic traffic control based on real (covering actual and history data) and simulated data (addressing future predictions).

The chat bot agents (as computational agents) can operate both in real and virtual worlds including games and providing a fusion of both worlds by seamless migration. Agents are loosely coupled to their environment and platform and interact with each other via Tuple Spaces (generative communication) and via uni-cast or broadcast signals. Mobility is provided by agent process snapshot migration between platforms.

The novel MAS crowd sensing and simulation framework, introduced in the next sections, is suitable to combine social and computational simulations with real-world interaction at run-time and in real-time, e.g., by integrating crowd sensing, using mobile agents. There are two classes of mobility that can be modelled in the simulation: (1) Social mobility on short- and long-term time scales (2) Goal driven traffic mobility.

Agent-based traffic management simulation still neglects social interactions and influence of social media, e.g., MATISSE, the Multi-Agent based TraffIc Safety Simulation System [11], which is a large-scale multi-agent based simulation platform designed to specify and execute simulation models for Agent-based Intelligent Transportation Systems. Therefore, a use-case is shown

evaluating simple social interaction and social structures combined with social and traffic mobility based on a parametrisable Sakoda model that is used to demonstrate the capabilities of agent-based crowd sensing integrated in and extending agent-based simulation. The crowd sensing performed by mobile agents is used to create digital twins of real humans (with respect to the social interaction model and mobility) based on individual surveys and user information collected via a chat bot dialog.

Among the capability of real-time simulation, the simulation framework is capable to create simulation snapshot copies that can be simulated in parallel in non-real-time to get future predictions (simulation branches) from actual simulation states that can be back propagated into the current real-time simulation world.

In a first simple demonstration we find that the new approach is a powerful tool and enables the study of large-scale social interactions. The strength of the framework lies in the potential inclusion of environment and real-life interactions.

In the future, we aim to use the framework in order to investigate different environments in quasi-experimental setups. Controlling all aspects of the virtual world allows us to explain the source of structural variation in human interactions, i.e., tackle the hard-to-observe mechanisms of micro-macro interactions. For instance, it is known that tie creation processes of social networks rely heavily on the (organisational) environment [12]. Utilising real-world interactions together with ABMs in carefully selected, systematically diverging environments creates a framework in which we can experimentally distinguish between ideal typical situations and understand how different structures result in similar/diverging behaviour. First applications might involve hierarchical/unstratified environments represented by different types of organisations. In so doing, we would add to the question how the same micro-mechanisms (e.g., homophily, balance) are created by and reproduce different network structures and, hence, extend emerging approaches on network ecological theory.

II. ARCHITECTURE

The proposed framework couples virtual and real worlds by integrating simulations with human interactions by using computational agents (chat bots) and physical behavioural agents inside the simulation.

The entire crowd sensing and simulation architecture consists of the following components:

1. Unified Agent Processing Platform based on JavaScript: JavaScript Agent Machine (JAM);
2. Crowd Sensing Software (Mobile App and WEB Browser using JAM);
3. Agent-based Simulation with Internet connectivity supporting two different agent types:
   - ***Physical behavioural agents*** representing individual artificial humans;
   - ***Computational agents*** representing mobile software, i.e., used for distributed data processing and digital communication, and implementing chat bots;
4. Chat dialogs, Chat bots and Mobile Agents collecting user and device sensor data;
5. Knowledge-based Question-Answer Systems, Natural Language Processing;

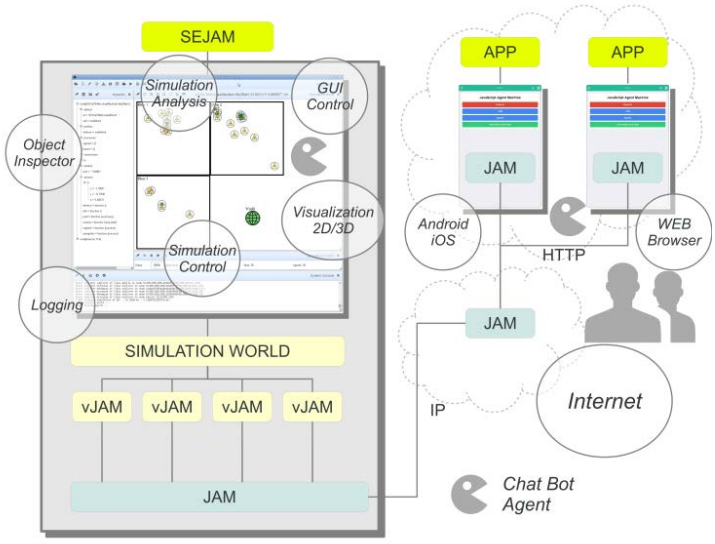

Fig. 2: Principle concept of closed-loop simulation for augmented virtuality: (Left) Simulation framework based on the JAM platform (Right) Mobile and non-mobile devices executing the JAM platform connected with the virtual simulation world (via the Internet) [6]

The mixed-model simulation world consists of physical and computational agents bound to logical (virtual) platforms (host of the agent) that are arranged or located on a lattice to provide world discretisation. The agents are mobile. Computational agents as mobile software processes can migrate between platforms (both in virtual and real digital worlds), whereas physical agents are fixed to their platform and only the platform is mobile (virtual world).

The agents are programmed in JavaScript and executed by the JavaScript Agent Machine (*JAM*) that can be deployed on a wide range of host platforms (mobile devices, servers, IoT devices, WEB browser [13]). JAM provides virtualisation by the Agent Input-Output System (AIOS), tuple spaces, and virtual (logical) nodes bound to a world contained in one physical JAM node. Each physical or logical JAM node can be connected with an unlimited number of remote JAM nodes by physical links (UDP/TCP/HTTP using the AMP protocol). Logical nodes can be connected by virtual links. Links provide agent process migration, signal (message) and tuple propagation.

Simulation is performed by the Simulation Environment for *JAM* (*SEJAM*) [14] [15], extending JAM with a simulation layer providing virtualisation, visualisation of 2D- and 3D worlds, and an advanced simulation control with a wide range of integrated analysis and inspection tools to investigate the run-time behaviour of distributed systems. Each *JAM* node is capable to process thousands of agents concurrently. *JAM* and *SEJAM* nodes can be connected in

clusters and in the Internet (of Things) enabling large-scale "real-world"-in-the-loop simulations with Millions of agents. Agents executed in the simulation have access to an extended simulation API. Physical agents can use an extended *NetLogo* compatible simulation API extension, too, enabling simulation world analysis and control, based on object iterators (*ask* and *create* statements, see [5]).

There is a significant difference between traditional closed-world simulations and simulations coupled with real world and real-time environments (human in the loop simulation). Closed simulations are performed on a short time-scale with pre-selected use cases and input data. The simulation can be processed step-wise without a relation to a physical clock. In contrast, open-loop simulation requires continuous simulation on a large time-scale creating Big Data volumes. A relation to a physical clock is required, too.

The mapping of physical on virtual worlds and vice versa is another issue to be handled. There are basically three possible scenarios and world mapping models (illustrated in Fig. 3):

1. The real and virtual worlds are isolated (no agent/human of one world knows from the existence of the other)
2. The real world is mapped on the virtual simulation world (simulating the real world with real and artificial humans or entities)
3. The virtual world extends the real world, e.g., by a game world, and real and virtual entities known from each other.

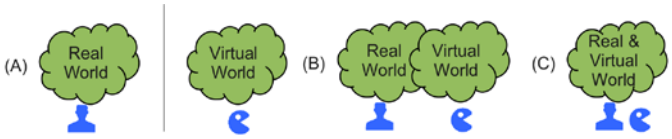

Fig. 3: (A) Real world only deployed with humans (B) Non-overlapping real and virtual world (C) Overlapping real and virtual world

III. WORKFLOW

The following Fig. 4 shows the principle work and data flow of the proposed architecture. The ABMS relies on generated data stored in a data base provided by Data Mining (DM) with sensor data from Crowd Sensing (CWS). performed by chat-bot agents. But the data flow is bidirectional, and agents can carry data generated by the simulation to mobile devices and users in the real world.

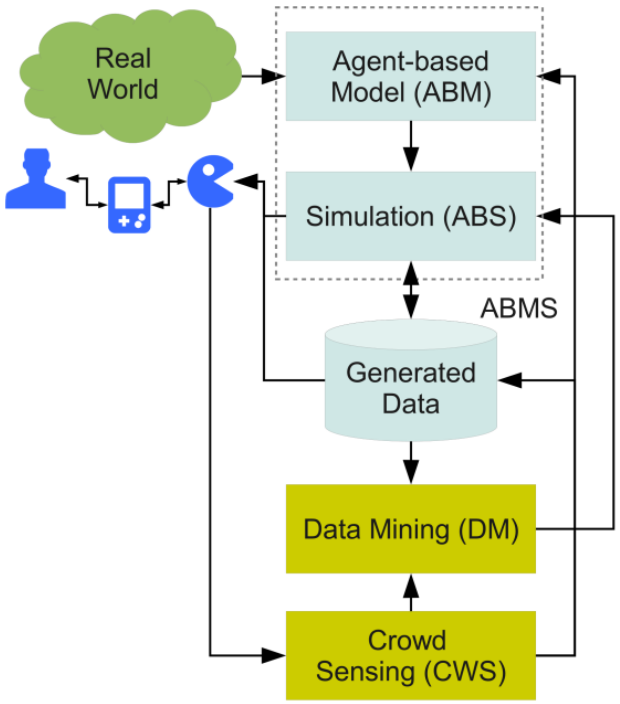

Fig. 4 Principle work and data flow integrating agent-based crowd sensing in agent-based modelling and simulation

IV. CHAT BOTS AS MOBILE AGENTS IN BOTH WORLDS

Mobile agents are used in this work for distributed data processing and data mining in the real and virtual simulation world seamlessly. It is assumed that the real world environments consist of communication access points (i.e., beacons, e.g., using WLAN, WWAN communication technologies) and mobile devices (smart phones, IoT devices). Commonly there is no globally visible organisational and network structure, i.e., agents cannot rely on a network world model to reach and identify specific devices. Each JAM node provides connectivity information, i.e., a list of all connected JAM nodes. Commonly, JAM nodes are connected peer-to-peer in mesh-like networks via TCP/HTTP or UDP connections.

Agents can communicate with each other either by exchanging tuples via a tuple space data base or by using signals (lightweight messages) that can be propagated remotely along agent migration paths. Tuple space access is generative communication, i.e., the lifetime of tuples can exceed the lifetime of the generating agent. Additionally, tuple space communication is data-driven and anonymously (sender and receiver need no knowledge about each other).

Mobile agents, e.g., chat bots, can be used to carry information from one location to another. Moreover, mobile devices carrying mobile agents can be used to collect, carry, and distribute data within large regions via tuple spaces, shown in principle in Fig. 5. The virtual simulation world is just another region in the mobility regions of agents, and

chat bots performing crowd sensing can migrate between real and virtual worlds seamlessly.

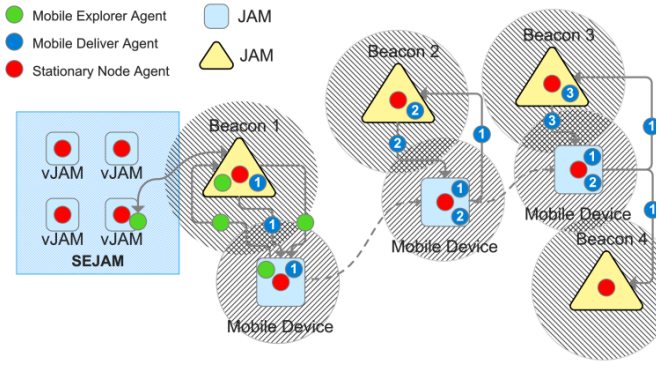

Fig. 5: Virtual simulator world connected to spatially distributed non-mobile beacons, mobile devices connected temporarily to beacons (cellular or local WIFI networks), and mobile agents (crowd sensing and chat bots) used for wide-range interaction

V. CROWD SENSING AND SURVEYS

The basic crowd sensing and social survey architectures can be classified in:

1. Centralised crowd sensing architecture with a single master instance (crowd sourcer)
2. Decentralised crowd sensing architecture with multiple master crowd sourcer instances
3. Self-organising and ad-hoc crowdsensing architecture without any crowd sourcer master instances

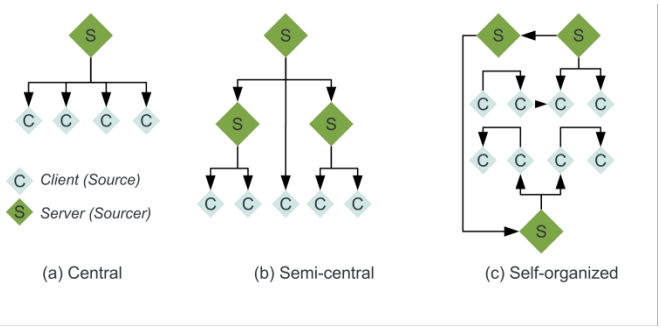

Fig. 6: Different crowd sensing architectures and strategies with data sourcer and data sources

The survey used in this work relates to the classical central approach, although the can be extended with a decentralised and self-organised approach, too, handled by the same framework and mobile chat bot agenzs. The survey aims to create digital twins in the virtual world from survey participants in the real world by deriving twin behaviour model parameters from the survey answers, discussed in the next section. The survey is performed by chat bot agents that can perform dynamic dialogs based on answers and context.

VI. CHAT BOTS AND HUMAN-AGENT INTERACTION

Mobile agents can migrate between different devices. The main goal of explorer agents is to collect crowd and social data. This data can be retrieved by device sensors (including aggregated virtual sensors) and information provided by users. Privacy and security have to be addressed. Among device sensor data like position, velocity, ambient light, which can be collected by mobile explorer agents automatically, user data can be gained by a human-agent dialog chat via a WEB Browser or Android/iOS App (see Fig. 7). Therefore, an explorer agent is capable to interact with humans by performing textual dialogs (question-answer surveys). Due to the distributed and parallel processing of loosely coupled agents filtering and scheduling of agent dialog request has to be performed. On each device participating in crowd sensing there is a mediator agent that manages and assess interaction requests from incoming (explorer) agents. Passed request are forwarded to a user-bot chat dialog. Responses to questions (requests) are passed back to the requesting explorer agents.

Each agent migrating to a device operated by humans (i.e., smart phones, car board computer, or fixed terminals) can ask questions via the APP tuple space by inserting a question tuple, finally handled by the node mediator agent. This question request tuple is evaluated and assessed by the mediator agent performing security and privacy checks, and finally passing the question to the chat blog waiting for human participation. The answer (if any) is passed back to the original requesting agent by an answer tuple, again previously checking privacy and security concerns. The principle format is shown below. The dialogs can be composed of dialog trees, i.e., questions and messages can be dynamic and depend on previously answered questions or environmental perception (sensors).

```
out(['Message',<id>,<message>])
out(['Question',<id>,<question>,
    {value: <default>,
     type:'number'|'string',
     icon:<icon-name>}]
inp.try(timeout,
    ['Answer',<id>,<question>,_],
     function (tuple) {
      if (tuple)
        this.result = tuple[3] })
```

In doubt, the question or answer is discarded or modified (e.g., filtered or annotated with additional warnings). The mediator agent performs question request scheduling to satisfy agent request and human interaction capabilities (high question rates will decrease the user interest and motivation to answer questions). Additionally, the mediator agent has to monitor chat bot dialogs by using natural language processing (NLP).

A typical chat dialog is shown in Fig. 7. It consists of question-answer snippets processed and controlled by the mediator agent operating on each device with user interaction.

The mediator agent can pass questions (or just standalone messages) to the user chat via a simple platform API

providing a question text, possible answers or input fields, and a callback function handling the result:

```
chat.message(<id>,<message>);
chat.question(<id>,<question>,{
     size: <test-size>,
     icon: <icon-name>,
     value: <default value>
     sub_type: 'number'|'string',
     placeholder: <placeholder-value>
 },function (result) {
   this.answer=result;
 },timeout);
```

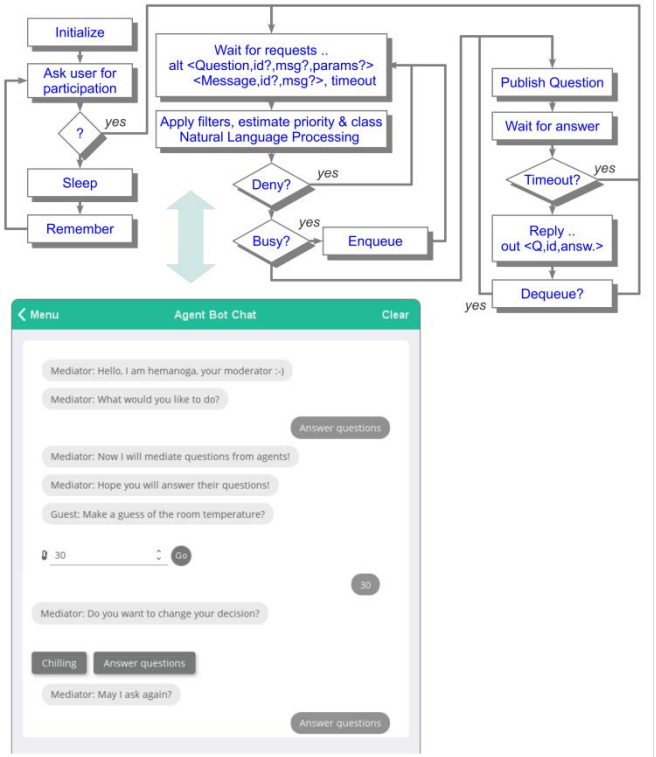

Fig. 7: (Top) Chat dialog of the JAM App showing messages and questions from chat bot agents and principle mediator agent behaviour scheduling the user chat (Bottom)

The following code describes the dialog script passed to a chat bot agent that is used to perform the crowd sensing on mobile devices by the chat bot agents. The question dialog is dynamic: (1) Some questions are only presented based on answers of previous questions → *cond* function (2) Some question texts are updated based on previous answers → *eval* function replacing text variables `$` with values returned by an array.

```
type dialog = {
   question:string,
   choices?:string [] | number [],
   value?:number|string,
   eval? : function (script) -> string,
   cond? : function (script) -> boolean,
   tag : string,
   answer?: string
} []
type script = dialog []
```

VII. Crowd Sensing and Cloning of Digital Twins

Commonly, simulation is performed with artificial agent models derived from theoretical considerations or experimental data. Augmented virtuality enables dynamic simulations with agents representing real humans (or crowds). By using crowd sensing it is possible to create digital twins of real humans based on a parameteriseable behaviour and interaction model. The parameters of artificial humans in the simulation represented by agents are collected by sensor data, i.e., surveys optionally fusioned with physical sensors like GPS. One simple example is shown in the next section undefined.

The crowd sensing via mobile chat bots enable the interaction of real humans with agents and digital twins in the simulation world in real-time and vice versa. The digital twins as well as the artificial physical agents in the simulation can interact by dynamically created (influenced) dialogs reflecting the state of the simulation world.

VIII. Social Interaction and Mobility by the Sakoda Model

Two demonstrate the augmented virtuality approach combining agent-based simulation with agent-based crowd sensing the *Sakoda* model was chosen as a simple social interaction and behaviour model between groups of individual humans posing self-organising behaviour (emergence) and structures of social groups by segregation. This self-organising behaviour can be observed on a long-time and long-range scale (classical segregation) and on short-time range and scale (e.g., city mobility).

The original Sakoda behaviour model [16] consists of social interactions among two groups of individuals evolving in a network according to specific attitudes of attraction, repulsion, and neutrality. An individual evaluates its social expectative at all possible available locations (starting at its current location), preferring originally those near individuals associated with attractive (positive) attitudes and avoiding locations near individuals associated with repulsive (negative) ones. This procedure is repeated randomly among all possible individuals; henceforth Sakoda's algorithm is iterated repeatedly developing a spatially distributed social system to an organised spatial pattern, although this depends on the parameter set of the model, crowd densities, and individualisation, introduced below.

For the sake of simplicity, there is a two-dimensional grid world that consists of places at discrete locations (x,y). An artificial agent occupies one place of the grid. Maximal one agent can occupy a place. The agents can move on the grid and can change their living position. It is assumed that there are two groups related to the classes *a* and *b* of individuals. The social interaction is characterised by different attitudes [16] of an individual between different and among same groups given by four parameters:

$$S = (s_{aa}, s_{ab}, s_{ba}, s_{bb})$$

The model is not limited to two groups of individuals. The *S* vector can be extended to four groups (or generalised) by the matrix:

$$S_{abcd} = \begin{pmatrix} s_{aa} & s_{ab} & s_{ac} & s_{ad} \\ s_{ba} & s_{bb} & s_{bc} & s_{bd} \\ s_{ca} & s_{cb} & s_{cc} & s_{cd} \\ s_{da} & s_{db} & s_{dc} & s_{dd} \end{pmatrix}$$

The world model consists of N places x_i. Each place can be occupied by none or one agent either of group α or β, expressed by the variable $x_i=\{0,-1,1\}$, or generalised $x_i=\{0,1,2,3,4,..,n\}$ with *n* groups. The social expectation of an individual *i* at place x_i is given by:

$$f_i(x_i) = \sum_{k=1}^{N} J_{ik}\delta_s(x_i, x_k)$$

The parameter J_{ik} is a measure of the social distance (equal one for Moore neighbourhood with distance one), decreasing for longer distances. The parameter δ expresses the attitude to a neighbour place, given by (for the general case of *n* different groups):

$$\delta_s(x_i, x_k) = \begin{cases} s_{\alpha\beta} & \text{, if } x_i \neq 0 \text{ and } x_k \neq 0 \text{ with } \alpha = x_i, \beta = x_k \\ 0 & \text{, otherwise} \end{cases}$$

Self-evaluation is prevented by omitting the current place (i.e., $k \neq i$). There is a mobility *m* factor giving the probability for a movement.

An individual agent ag_i of any group α (class from the set of groups) is able to change its position by migrating from an actual place x_i to another place x_m if this place is not occupied ($x_m=0$) and if $f_i(x_m) > f_i(x_i)$ and the current mobility factor m_i is greater than 0.5. Among social expectation (resulting in segregation), transport and traffic mobility has to be considered by a second goal driven function $g(x_m)$, commonly consisting of a destination potential functions with constraints (e.g., streets). If $g_i(x_m) > f_i(x_m) > f_i(x_i)$ than the goal driven mobility is chosen, otherwise the social driven.

$$g : (x_i, x_m, x_k, v, t) \rightarrow \mathbb{R}$$

The mobility function *g* returns a real value [0,1] that gives the probability (utility measure) to move from the current place x_i to a neighbour place x_m to reach the destination place x_k with a given velocity *v*. The *g* function records the history of movement. Far distances from the destination increases *g* values. Longer stays at the same place will increase the *g* level with time *t*. Social binding (i.e., group formation) will be preferred over goal driven mobility.

The computation of the neighbouring social expectation *f* values is opportunistic, i.e., if *f* is computed for a neighbouring node assuming the occupation of this neighbour place by the agent if the place is free, and the current original place is omitted x_i for this computation. Any other already occupied places are kept unchanged for the computation of a particular *f* value. From the set of neighbouring places and their particular social and mobility expectations for the specific agent the best place is chosen for migration (if there is a better place than the current with the above condition). In this work, spatial social distances in the range of 1-30 place units are considered.

Originally, the entire world consists of individual agents interacting in the world based on one specific set of attitude parameters *S*. In this work the model is generalised by assigning individual entities its own set *S* retrieved from real humans by crowd sensing, or at least different configurations of the *S* vector classifying social behaviour among the groups. Furthermore, the set of entities can be extended by humans and bots (intelligent machines) belonging to a group class, too.

Segregation effects inhibit individual movement until a different social situation enables a movement. Transportation mobility triggers movement even if there is no social enabler. This reflected in the extended Sakoda model by the mobility factor *m* and the goal driven expectation function *g* that control mobility and overlays social and transportation and traffic mobility. The mobility function *g* includes random walk and diffusion behaviour, too. Constrained mobility is one major extension of the original Sakoda model presented in this work.

IX. Model Parameters and Crowd Sensing

Creation of virtual digital twins is the aim of the crowd sensing. The crowd sensing is performed with chat bot agents. One stationary agent is operating on a user device, e.g., a smart phone, and another mobile agent is responsible to perform a survey (either participatory with a former negotiation or opportunistic ad-hoc). The results of the survey, a set of questions, are used to derive the following simulation model parameters:

```
parameters = {
    group : string "a"|"b"|..,
    social-distance: number [1-100],
    mobility-distance: number [1-100],
    social-attitudes : [saa,sab,sba,sbb] |
                       number [][],
    mobility : number [0-1],
    position : {x:number,y:number},
    destination : {x:number,y:number}
}
```

The *group* parameter sorts the user in one of two classes *a*/*b*, the *social-distance* parameter is an estimation of the social interaction distance, the *social-attitudes* parameter is the *S* vector, but limited to a sub-set of all possible *S* vector combinations (discussed below), and the *mobility* parameter is a probability to migrate from one place to another. The position (in cartesian coordinates) is derived from the living centre of the user (global position data, GPS) and mapped on the simulation world (*x*,*y*). The *S* vector parameter determines the spatial social organisation structure. Typical examples of the *S* vectors with relation to social behaviour are [16]:

- (1,-1,-1,1): Typical segregation with strong and isolated group clusters
- (0,-1,-1,0): Mutual suspicion
- (1,-1,1,-1): Social climbers
- (1,1,-1,-1): Social workers
- (1,1,1,1): Inclusion
- (1,0,-1,1): Traffic (a:cars, b:bicycles)

The crowd sensing extends the simulation with the following dynamics and changes:

1. Enhancement of the synthetic simulation with real world data by digital twins not conforming to an initial parameterisation of the artificial individuals (affecting *S* vector, spatial social interaction distance, mobility)
2. Adaptation and change of the fraction of different groups (commonly a equally distributed fraction from each group is assumed)
3. Convergence and divergence of the emergent behaviour of group formation (spatial organisation structures)
4. Influence of the real world users by chat bots with data from the virtual simulation world

The dynamics in the simulation world can be back propagated to the real world by chat bot agents, too, delivering information and opinions formed by the artificial entities. Among classical surveys chat bots can perform manipulation by distributing biased information or opinions via the chat dialog interface.

X. Simulation Model and Architecture

In this work, the *SEJAM* simulator is used to create a simulation world (consisting of individuals represented by agents) that is attached to the Internet enabling remote crowd sensing with mobile computational agents. The *SEJAM* simulator is basically a graphical user interface (GUI) on the top of the JavsScript Agent Machine (*JAM*), shown in Fig. 2. *JAM* agents are programmed in JavaScript with an Activity-Transition graph (ATG) behaviour model composing the agent behaviour of activities performing actions (computation, interaction, mobility). A simulation world consists of multiple virtual *JAM* nodes. In contrast to pure ABM simulators like *Netlogo*, the *SEJAM* simulator simulates computational agents and agent processing platforms and is therefore primarily an ABC simulator, but can be used for ABM like simulations, too. The simulation model and architecture consists of agents and nodes that can execute and control agents. Computational agents are always bound to a virtual *JAM* node. A node can be created dynamically at simulation run-time or during the initialisation of the simulation world. A node is related to a virtual position in a two-dimensional world (similar to a patch in the *Netlogo* world model). In contrast to a *Netlogo* simulation model using one main script describing the entire world, agents, and resources, a dedicated world agent (operating on a dedicated world node) controls the simulation in *SEJAM*. A node position can be changed by agents (related to the movement of turtles in the *Netlogo* model. *JAM* agents play two different roles in the simulation: (1) ABC: Mobile Crowd Sensing (2) ABM: Representation of humans, bots, and digital twins of humans. The *JAM* agent behaviour model bases on activity-transition graphs partitioning the behaviour in activities performing actions (computation, mobility, interaction with other agents and the world). An activity is related to a sub-goal of a set of goals of the agent. Modification of agents (visual, shape, position, body variables) can be formed globally by the world agent via a simulation interface providing a *NetLgo* API compatibility layer, or by the individual agents.

XI. Comparison of NetLogo and SEJAM Simulations

The well known and established *NetLogo* simulator is (commonly) used for the ABM domain only. In contrast, the *SEJAM* simulator is primarily used for the ABC domain, although any physical simulation can be transformed in a computational task implementing the behaviour of a physical entity. The fusion of real worlds deploying computational agents (that interact with machines and humans) with virtual simulation models requires a mapping methodology to be able to simulate physical agents (i.e., artifacts of real entities) using computational agents.

Computational agents as mobile software processes require a connected communication network of host platforms (computers) to migrate along a path *AB*. A human being, in contrast, do not require such a digital transport network. In a simulation world, a physical (pure behavioural) agent can overcome arbitrary distances in one step (in principle).

To combine social interaction and computational simulation models, the *SEJAM* simulation model was extended by two super classes of agents: (1) Computational (2) Behavioural/Physical agents. A behavioural agent consists of a *<mobile node, physical agent>* tuple. The behavioural agent is bound to this node and can change its position only by moving the node carrying the agent. In contrast, computational agents are mobile and can hop from one platform to another if there is a communication link between both platforms by performing process serialisation and deserialisation. The link can be virtual (inside the simulation world) or physical connecting a logical node of the simulation world with another *JAM* platform in the real world (e.g., a smartphone) via the Internet.

The mapping of *NetLogo* model constructs and statements on the *SEJAM* simulation model basically consists of an simulation API extension. There are two approaches implementing the *NetLogo* patch grid world in the *SEJAM* simulation world: (1) Floating grid with mobile nodes carrying immobile agents representing turtles (active agents) and resources representing patches (2) Static grid with immobile nodes and mobile agents (turtles) representing patches. Only the first approach is considered (although both can be implemented with *SEJAM*). In *NetLogo*, the agent behaviour is entirely controlled and executed by a global observer (centralised macro control), whereas in *SEJAM* the agent behaviour is controlled and executed by each individual agent (decentralised micro control).

XII. Experiment and Evaluation: Use-case Smart City

The experiment aims to study crowd formation in cities (places, streets, buildings) based on physical and virtual social interaction via social media and opinions and was performed with an artificial map of a city (simplified real world). Distributed crowd sensing via chat bots introduce updates and disturbance in the social formation structures. The mobility of physical agents are modelled by the extended Sakoda model introduced in Sec. undefined and depends on local social interaction (temporary crowd formation) and a potential field attraction regarding the destination in the city that has to reached. Mobility is spatially constrained (streets).

The crowd sensing tries to estimate local people movement and compound crowd formation constrained by urban structures (like streets) based on an estimated individual *S* matrix. For the sake of simplicity, two groups are assumed (*a*,*b*) differing in attitudes and behaviour.

The two groups, for instance, could be car and bicycle users acting as public traffic participants, differing in social, mobility, and group formation behaviour. The survey can be opportunistic, e..g., ad-hoc and occasional with a situation-aware dialog for specific traffic situations, or participative with a more general survey character.

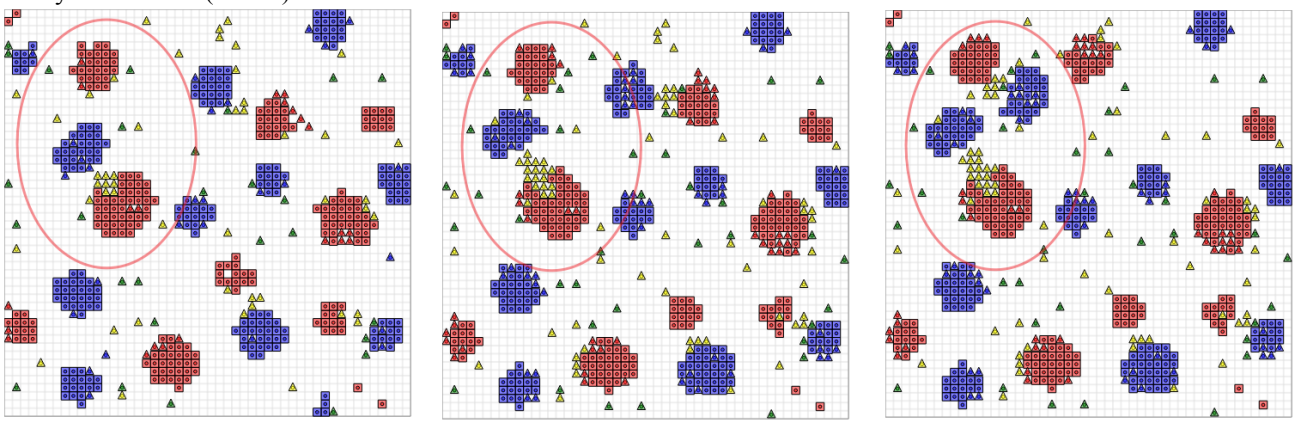

Fig. 8: Simulation world with long-term mobility, without spatial and context constraints at different simulation times (500/1000/1500 steps) consisting of 200/200 a/b class agents (blue/red squares) all with S=(1,-1,-1,1) and r=3 parameter settings and additionally up to 200 digital twins (triangles with colour based on individual S/r parameters)

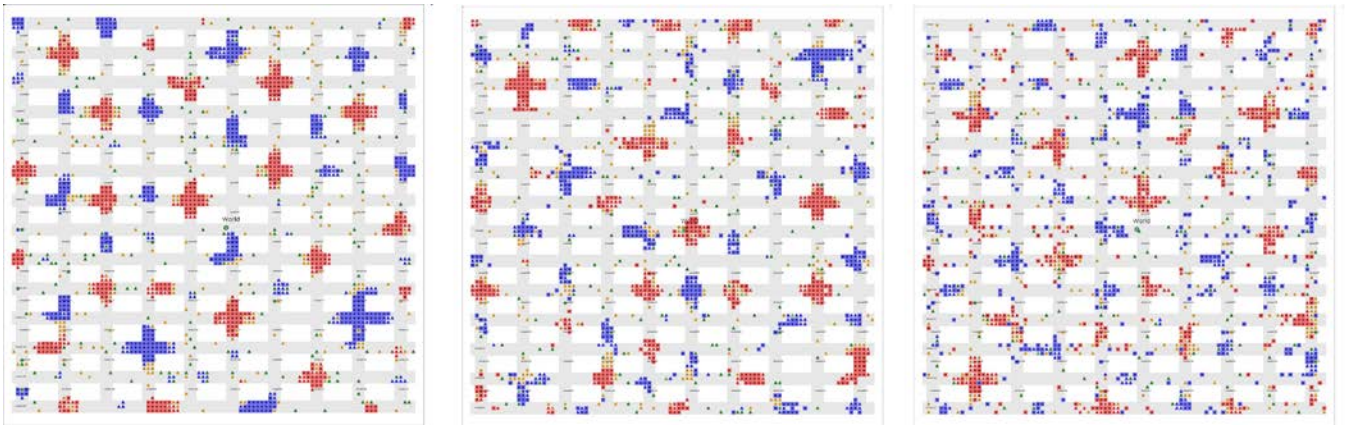

Fig. 9: World consisting of mobility constraints (streets), 400/400 class a/b agents (blue and red squraes) and digital twins (triangles) with different model parameters. (Left) Only social driven mobility and clustering (Middle) Social and traffic driven mobility with low attraction of the destination (Right) Social and traffic driven mobility with high attraction of the destination

The position of the digital twins created in the two-dimensional simulation world is estimated by GPS and IP localisation collected by the chat bot agent.

The initial simulation was performed with an unified *S=(1,-1,-1,1)* setting for all agents (blue and red squares) of both classes and unconstrained mobility leading to classical segregation structures (strong isolated clusters), shown in Fig. 8. Digital twins (triangles) retrieved form crowd sensing surveys (chat dialogs) were added to the simulation dynamically. The *S* vector and the social distance *r* of digital

twin now depend on the answers given by the (real) humans, which can differ from the initial *S* setting. These agents (if their *S* differs from the basic model) create a disturbance in the segregation patterns. Agents with $s_{\alpha\beta} = 0/1$ and $s_{\alpha\alpha} = 0/1$ can be integrated in both groups and are able to bind different groups close together (see the development and movement of the blue and red clusters inside the red circle in Fig. 8).

Using an initial unified stable parameter set for all agents, digital twins with varying and different parameter sets based on survey data can be used to test the stability and convergence criteria of structure formation or mobility patterns like in traffic management on a fine grain scale.

In a second run. mobility constraints by streets were added, shown in Fig. 9. Mobility of agents were driven by reaching a destination in the simulation world (either chosen randomly or by user information in the case of digital twins) and damped or delayed by social interaction, depending on individual *S* vector, social and mobile interaction distances. The simulation results (snapshots after 3500 simulation steps) show an increasing disturbance of social aggregation patterns resulting from social interaction (mobility due to social attraction) by goal driven traffic mobility (attraction by destination). Again, $s_{\alpha\beta} = 0/1$ and $s_{\alpha\alpha} = 0/1$ twins act as compound glue and brings homophobic groups closer together. With increasing influence of goal driven traffic mobility over social attraction and mobility the social clustering gets more fuzzy and diffuse, shown on the right simulation snapshot of Fig 9.

The recognition of flows and structures (clusters) in the simulation can be finally used to manage real world traffic by agents in real-time. Real-world traffic flows can be influenced either by technical traffic infrastructure, by traffic information systems, or by influencing people decision making (on a temporally and spatially short-range scale).

XIII. CONCLUSIONS

The MAS simulation framework, introduced in this work, provides the bidirectional integration of Data Mining (DM) via Crowd Sensing and Agent Based Modelling and Simulation (ABMS) (i.e. applying DM in ABMS by creating generated simulation data and applying ABMS in DM by back propagating generated simulation data). The framework is suitable to combine social and computational simulations with real-world interaction at run-time by integrating crowd sensing, using mobile computational agents. The *JAM* agent processing platform can be deployed in heterogeneous networks on a broad range of devices including simulation software. This is enabled by programming agents in JavaScript. Chat bots with dynamic (situation dependent) dialogs can be implemented with *JAM* agents directly using a unified dialog structure and API.

A simple use-case demonstrated social interaction and social structure formation based on a parametrisable Sakoda model extended with digital twins retrieved by agent-based crowd sensing integrated in and extending agent-based simulation. Th social models overlays social and traffic mobility. The crowd sensing performed by mobile agents is used to create digital twins of real humans (with respect to the social interaction model and mobility) based on individual surveys via a chat bot dialog.

The survey data collected by the chat bot agents enable the classification of humans and other intelligent chat bots by means of the *S* vector derived from the answers. Chat bots are the link between virtual and physical worlds.

In the future, we aim to use the framework in order to simulate different environments in quasi-experimental setups. Controlling all aspects of the virtual world allows us to explain the source of structural variation in human interactions, i.e., tackle the hard-to-observe mechanisms of micro-macro interactions.

REFERENCES

[1] H. Hamidi and Ali Kamankesh, *An Approach to Intelligent Traffic Management System Using a Multi-agent System*, Int. J. ITS Res. (2018), vol. 16, pp. 112-124

[2] F.-Y. Wang, *Agent-Based Control for Networked Traffic Management Systems*, Intelligent Transportation Systems, no. September/October, 2005.

[3] O. Baqueiro, Y. J. Wang, P. McBurney, and F. Coenen, *Integrating Data Mining and Agent Based Modeling and Simulation*, in *ICDM 2009, LNAI 5633*, 2008.

[4] N. Gilbert, *Agent-based social simulation: dealing with complexity*, 2004.

[5] U. Wilensky and W. Rand, *An Introduction to Agent-Based Modeling Modeling Natural, Social, and Engineered Complex Systems with NetLogo*,. MIT Press, 2015.

[6] S. Bosse, U. Engel, *Augmented Virtual Reality: Combining Crowd Sensing and Social Data Mining with Large-Scale Simulation Using Mobile Agents for Future Smart Cities*, Proceedings, Volume 4, ECSA-5 5th International Electronic Conference on Sensors and Applications 15–30 November, 2018 DOI 10.3390/ecsa-5-05762

[7] T. Calenda, M. D. Benedetti, F. Messina, G. Pappalardo, and C. Santoro, *AgentSimJS: A Web-based Multi-agent simulator with 3D capabilities*, 2016.

[8] R. K. Ganti, F. Ye, H. Lei., *Mobile Crowd Sensing: Current State and Future Challenges*. IEEE Communications Magazine, vol. 49, no. 11 (2011)

[9] K. H. van Dam, I. Lukszo, N. Zofia. 2013. Eds., *Agent-Based Modelling of Socio-Technical Systems*. Springer Berlin

[10] T Leppäne, J. Á. Lacasia, Y. Tobe, K. Sezaki, and J. Riekki. 2017. *Mobile Crowd Sensing with mobile agents*. Autonomous Agents and Multi-Agent Systems, vol. 31, no. 1, pp. 1-35

[11] M. Al-Zinati, R. Wenkstern, *An agent-based self-organizing traffic environment for urban evacuations*, Proceedings of the The Sixteenth International Conference on Autonomous Agent and Multiagent Systems, AAMAS '2017, Sao Paulo, Brazil, May 2017.

[12] D. A. McFarland, J. Moody, D. Diehl, J. A. Smith, R. J. Thomas, *Network Ecology and Adolescent Social Structure*, American Sociological Review, 2014, http://dx.doi.org/10.1177/0003122414554001

[13] S. Bosse, *Mobile Multi-Agent Systems for the Internet-of-Things and Clouds using the JavaScript Agent Machine Platform and Machine Learning as a Service*, in The IEEE 4th International Conference on Future Internet of Things and Cloud , 22-24 August 2016, Vienna, Austria, 2016, DOI: 10.1109/FiCloud.2016.43

[14] S. Bosse, *Smart Micro-scale Energy Management and Energy Distribution in Decentralized Self-Powered Networks Using Multi-Agent Systems*, FedCSIS Conference, 6th International Workshop on Smart Energy Networks & Multi-Agent Systems, 9-12.9.2018, Posznan, Poland, 2018

[15] S. Bosse, E. Pournaras, *An Ubiquitous Multi-Agent Mobile Platform for Distributed Crowd Sensing and Social Mining*, FiCloud 2017: The 5th International Conference on Future Internet of Things and Cloud, Aug 21, 2017 - Aug 23, 2017, Prague, Czech Republic

[16] P. Medina, E. Goles, R. Zarama, and S. Rica,*Self-Organized Societies: On the Sakoda Model of Social Interactions*, Complexity, 2017.