Finding a Structure: Evaluating Different Modelling Languages Regarding Their Suitability of Designing Agent-Based Models

Poornima Belavadi(✉), Laura Burbach, Martina Ziefle, and André Calero Valdez

Human-Computer Interaction Center, RWTH Aachen University, Campus Boulevard 57, 52076 Aachen, Germany
{belavadi,burbach,ziefle,calero-valdez}@comm.rwth-aachen.de

Abstract. Several approaches to standardize the creation of agent-based models exist, but there is no perfect way to do it yet. In this study we analyze, whether two modelling languages (i*Star, UML) can help in designing agent-based models. We identified requirements for building agent-based models and analyzed to what extent the requirements can be met by applying modeling languages. We reflect whether the application of modeling languages can profitably facilitate the creation of agent-based models. We found that modeling languages can meet some requirements for creating agent-based models. Finally, modeling languages offer an added value to the creation of agent-based models, but their application also requires more time than creating a model without their application. However, when creating agent-based models, a considerable amount of time should be spent to decide what the model would depict. Our approach can be helpful in the future for creation of agent-based models.

Keywords: Agent-based modelling · Modelling languages · Programming languages · Simulation · NetLogo · UML · i* · i*Star

1 Introduction

Agent-based models have been built for many decades [32], but have only recently been applied in the social and social-ecological sciences [37]. For most of the time, the process of development has been highly individual and has not followed standardized criteria. To facilitate **reuse and replication**, it would be desirable to use **standardized approaches in agent-based modeling** that allow describing models as concisely and completely as math can be described in the universal language of mathematics [17].

As it is not always clear to the developers of agent-based models at what level of detail the models should be described and where, how, and what kind of information should be given, **many descriptions are difficult to read**.

V. G. Duffy (Ed.): HCII 2021, LNCS 12777, pp. 201–219, 2021.
https://doi.org/10.1007/978-3-030-77817-0_16

Similarly, it is often difficult to replicate a published model, which is complicated by the fact that agent-based model descriptions are often incomplete [17].

For this reason, Grimm et al. [16,18] designed the ODD protocol. They believed that many **features of agent-based models are common** and thus enable a common language. The acronym stands for Overview, Design concepts, and Details and the protocol is intended to serve as a standard form for describing agent-based models.

In this article, we reflect on the suitability of different approaches to build agent-based models. To date, agent-based models are most commonly created using NetLogo, which is a simulation environment. In addition, other models have been created in general-purpose programming languages (such as Java, Python or Julia). The **ODD protocol provides a standardized process** towards creating agent-based models using guideline questions. We examine whether it makes sense to **additionally use one of the two class languages UML and i*Star** (actually written i* and pronounced i-Star) when creating agent-based models.

2 Related Work

We first show some fundamental underlying aspects of agent-based modeling: Emergence, complexity vs. simplicity, and realism. Next, we introduce the method agent-based modelling. Then, we show how models are created (simulation environment, programming language, ODD-protocol). Last, we introduce the modelling languages UML and i*Star.

2.1 Aspects of Agent-Based Modelling

Agent-based models simulate individuals or agents whose behavior is guided by simple rules. The behavior and properties of the agents are described at the micro-scale, revealing complex behavior at the macro-scale [31].

Emergence. The whole is more than the sum of its parts. This statement points to a strength of agent-based models. We normally look at individual subcomponents of a system and infer the behavior of the whole system from them, the overall behavior does not simply result from just observing the components. Moreover, it is often difficult to look at the overall behavior. Instead, it is easier to look at individual behavior. **Emergence refers to systems that are not merely the sum of the individual components**, but where the individual components complement and influence each other, resulting in hard-to-predict systems.

With agent-based models we can create individual agents and shape their behavior to follow individual rules at micro level. Agents reside in an environment and interact with both the environment and other agents, creating hard-to-predict social patterns. We can thus see emergence or emergent behavior [5].

Complexity, Simplicity, and Realism. Complex systems consist of different ontological levels, which can also be considered as interacting subsystems and take place on a micro and macro level [8]. To understand why a system is complex, it is important to look at how the system is (structurally) built [36]. Systems can be both complex and complicated, but they are not always both. Both terms refer to a system that is made up of components, but they refer to different aspects of the system. If there are many subcomponents that affect each other and it is therefore difficult to predict the system behavior, the system is complex. **Complexity depends on the structure of interactions and the underlying dynamics** of a system [6,36,38]. A system is merely complicated if it is difficult for us humans to understand, such as a mathematical equation [6,38].

Using simulations we can analyze complex systems well. Modelling the individual parts of a system makes the overall behavior visible [12].

2.2 General Idea of Agent-Based Models

In agent-based models, agents can be in various forms (such as individuals, trees, atoms) and an environment in which the agents move [4]. Agent-based models also have a topology, which indicates how the agents are connected to each other.

Agent-based models represent a way of thinking rather than a technology [4]. Agent-based models cannot represent reality and they are never fully realistic. As is common for models, they depict reality in a simplified way. They allow to observe emergent behaviors in complex systems by mapping simple rules. Thus, it is possible to **observe and understand the behavior of complex systems without knowing the entire complex system**. Agent-based models are used to analyze systems and patterns from the bottom up. In this process, the agents exhibit heterogeneous behavior, which means that different agents exhibit individually different behavior. Furthermore, agent-based models contain stochasticity, meaning they can vary randomly [20].

In agent-based models, the agents are created programmatically as a template. They move in the environment and make decisions in the simulation depending on how they perceive that environment. Their perception influences their behavioral intent. Often, **agents are connected to other agents through a network**. The network then usually ensures that the agents influence each other.

The modeler can decide whether **agents behave deterministically or randomly**. Random behavior can be used, when not all aspects of the model need to be specified to answer the research question. This reduces complexity and the model can still represent behavior approximating real world concepts [39].

2.3 The ODD Protocol

Many have criticized that it is difficult to relate and build upon existing data, methods, and models. This has led scientists to develop several approaches in parallel to address the *replication crisis* [13,27,28] One of these approaches is

the *ODD protocol*, which was developed in 2006 [16] and extended in 2010 [18] and 2020 [17] and has already been used by a lot of modelers (e.g. [29]).

The *ODD protocol* (*Overview*, *Design concepts*, *Details*) is intended to **provide guidelines for writing and reading agent-based models** and thus facilitate the standardized creation of models (see Fig. 1). The three categories *overview*, *design concepts*, and *details* are further divided into seven elements. First, an *overview* of the model is provided. *Design* shows how the important design concepts to the creation of agent-based models have been implemented. *Details* look at all the other details of the model [17].

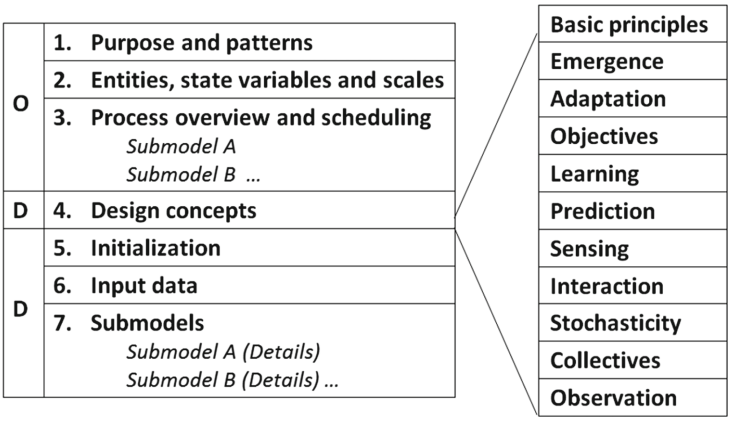

Fig. 1. ODD protocol: structure of model description; used from Grimm et al. [17]

Eleven different *design concepts* can be described. If *design concepts* do not apply, they can be omitted. Modelers should consider all 11 *design concepts* and particularly should decide **which key processes** in their model **are governed by empirical parameters and rules, and which processes are characterized by agents making adaptive decisions**. Typically, only a few processes can be adaptive, so it is important to consider and justify for which processes adaptive adaptation makes the most sense [17].

Through the *ODD protocol*, requirements are placed on modelers: They should describe their models in detail. They should specify the individual parts of their models and, in particular, provide the *design concepts*, which are unique to agent-based models. Through these requirements, modelers must also **reflect, explore, investigate, and justify the parts of the model design** [30].

Even though many modelers from different disciplines have used the *ODD protocol*, **still many papers about agent-based models are published without the use of it** [17]. As the description using the *ODD protocol* often becomes longer than describing the model without the protocol, it discourages some modelers from using it. In addition, there are specification languages that are more concise than *ODDs* (e.g., Z notation; [40]). However, it is difficult to read these specification languages if one is not familiar with them.

Reading the computer code of the model is a good way of understanding the workings of the model and replicating it. In contrast, *ODD* is more narrative, which makes many modelers feel it is less accurate and complete. In their view, *ODD* provides less opportunity to reproduce the model [2]. On the other hand, computer code can also be difficult to understand. Grimm et al. [17] recommend that to understand models accurately, modelers might consider an *ODD* in combination with the model code if they find the written description of the *ODDs* ambiguous [17].

2.4 Modelling Languages

Agent-based modelling has become a popular method for studying complex phenomena. However the available methods to design agent based models are not suitable for non-agent experts [19]. **Modelling languages are used to help in the design and construction of specific components or parts of a system** by following a systematic set of rules and frameworks. They can be either textual or graphical. In this section, we describe process modelling and goal modelling languages and their suitability for the research questions (see Sect. 1) we answer in this paper.

Process Modelling Languages. Process modelling languages are used to depict an actual process in an abstract manner by selecting the process elements that are considered important for the model's purpose. A Process model provides guidance for building a system by breaking down the process description into sufficient details [9]. Humphrey and Kellner [21] define a process as "a set of partially ordered steps intended to reach a goal." A *process element* is any component of a process and a *process step* is an individual or atomic action in a process. A process model is made up of individual model elements: *agent*, *role*, *artefact*. An *agent* performs the process element. An ordered set of process elements grouped together and assigned to an agent form a *role*. The outcome or the product created or modified by performing the process is an *artefact*. **Process modelling focuses on the interacting behaviour of agents** [9]. Issues handled by process models range from understanding a process to performing the process.

Some of the important characteristics of process modelling languages include modelling support, which takes care of all the process elements and communication between different agents and parallel activities. They also provide support to start a new task asynchronously, keep track of the unfinished tasks and provide visual notations for better understanding of the process by humans [43]. A process model provides **several perspectives to the underlying process**: In the *functional* perspective, the model focuses on *what* process elements are being executed and what are the relevant data and other entities required. *Behavioural*, focuses on *when* the process elements are executed and *how* they are done. *Organisational*, focuses on *where* and by *whom* in the organization are the processes being performed. *Informational* is related to the entities that are

produced as a result of one or more process actions. A process modelling language focuses on one or more of these perspectives [9]. ***Unified Modelling Language (UML)* is a popular process modelling language** that we consider in this paper to be suitable for designing ABMs.

Goal Modelling Languages. Goal modelling languages help in understanding the requirements of a system. Goal modelling assists in **identifying the system's goals** by asking questions like *what the system needs to do?*, *why a particular functionality is needed?* and *how it can be implemented?* [26]. Goal modelling has gained popularity because of its efficiency in Requirements Engineering (RE). Goal modelling not only helps in elaborating the requirements of the system but also in validating the completeness and correctness of the requirements [22].

There are a lot of goal modelling languages in the literature, however there is no standard construct for them as each language offers their own syntax, semantics and process. This is reported to be the main reason for these languages not being widely accepted [25]. This has motivated the research on goal modelling languages for understanding the effectiveness and efficiency to solve modelling problems. In a typical simulation, software agents are independent entities that interact with one another, cooperate and coordinate to achieve goals. Goal oriented languages are also well suited for agent based simulations. **Many goal modelling language constructs focus on agent-based simulation concepts** and a lot of research has gone into understanding the effectiveness of these languages for modelling agent-based systems [26]. Therefore, we further look into i*Star goal modelling language in this paper, for its suitability in designing ABMs.

3 Requirements Identification

In this study, we first considered different ways to design agent-based models. Second, we identified requirements that are necessary to design agent-based models. Third, we looked at questions each type of modelling language answers.

3.1 Different Ways to Design Agent-Based Models

There are **several ways to design an agent-based model:** One option is to build on existing theory—to evaluate, whether the assumptions from the theory also apply to the context under consideration or—to reflect, whether the theory needs to be updated. Besides, an agent-based model can be based on theoretical assumptions and empirical results as well. Subsequent to a profound literature review, for example an online survey can be conducted and the agent based model can be based on the survey results. This enables to measure theoretical or new assumptions using a specific sample. This ensures, that the agent-based model built on these results can be—at least for the considered sample—more realistic. Models built using either of these approaches can be additionally verified against real data (e.g.: data collected from social media).

3.2 Requirements of Agent-Based Models

In this section **we identify the necessary requirements when designing agent-based models**. Among other things, we have oriented ourselves on the steps and components of the *ODD protocol* [1], as well as adding our own considerations. A list of all collected requirements can be seen in Table 1.

To create an agent-based model, we first need a *suitable idea*, what we want to simulate—a research question—to be analyzed using agent-based modeling. We must reflect, why agent-based modeling is a well-suited method to our question and which *added value* a simulation offers compared to other empirical methods.

For other modellers to understand our model, they must understand the *model's purpose*. For this, **a general purpose of the model should be determined before determining a more specific purpose**. General purposes can be *prediction*, *explanation*, or *description* of the model. To keep the purpose of the model specific, it should state what components the model includes or what is not included: The goal is that **all entities and mechanisms contained in the model are necessary to achieve its purpose**. To avoid imprecise purposes, they should be deliberately phrased as a question (e.g., *do mechanisms A, B, and C explain observed phenomena X, Y, and Z*) [7].

One added value of agent-based models can be that it is possible to observe and analyze an *emergent aspect* (see Sect. 2.1). Therefore, when designing a model, it might be useful to **think about what potential unexpected outcomes might appear** in the simulation runs. Likewise, we can think of **potential *dynamics* or *dynamic processes*** in the simulation.

Different Ways to Design Agent-Based Models. As mentioned in Sect. 3.1, there are different *ways to design agent-based models* and we need to decide, on which we want to base our agent-based model. First, we need to identify *theoretical assumptions* addressing our research question/interest. Then, we can decide, whether an *empirical analysis* (e.g.: online survey) is needed to get a more realistic simulation (see Sect. 3.1). For empirical analysis, we consider **which processes in the simulation should be based on empirical results**. Besides, we check the possibility of *verifying the simulation against real data* and if so, which real data to use and which parts of the simulation can be verified using the data.

Adaptive Key Processes. As described in Sect. 2.3, we must also **decide *which key processes* in our model *should be adaptive*** versus based on empirical parameters and rules. For that, we need to reflect and reason, why we have chosen this process(es) to be adaptive.

Contexts Well-Suited for Agent-Based Models. We can also think of *contexts well-suited to agent-based models* or questions that lend themselves to be examined with an agent-based model. Agent-based models are particularly useful, if the focus is on the question of **how a system can adapt to changing conditions (robustness)**. Besides, they are useful, when we consider complex

problems. Another suited use case for agent-based models are simulations that involve emergence (see Sect. 2.1). Also a form of interaction, social influence, learning, or dynamics can be considered well in agent-based models.

Entities, State Variables, and Scales. The third element of the *ODD protocol* describes three interconnected parts of an agent-based model. The first part considers *entities*, independent objects or actors behaving as a unit. Entities can be *spatial entities* (e.g., grid cells or GIS polygons), *individual agents* and *collectives* (groups of agents with the same behaviors and attributes) [1]. So, when designing an agent-based model, it is also important to consider ***what types of entities* the model contains** (see Table 1), ***which entities* occur** in the model and ***why they occur***, as well as ***how many types of entities*** occur. Each type of entity should be described separately. In Table 1, we listed what should be considered when describing *agents*. The *environment entity* describes how other entities perceive the environment and specifies the time of the simulation [1].

The second part of the third *ODD element* describes the *state variables* or *attributes* for each type of entity. **The state variables indicate *what state an entity is currently in***. The state variables distinguish an entity from other entities of the same type. The *state variables* can also indicate how an entity changes over (the simulated) time. The characteristics of the *state variables* like *what the variables represent* and *what unit they have* should be explained while defining the *state variables*. Furthermore, **variables can change over time and be dynamic or static**. The type of variable (*integer*, *floating point number*, *text string*, *coordinate set*, *Boolean (true/false) value*, *a probability*) and the range they encompass should also be considered.

In addition, the *spatial* and *temporal scale* of the model should be described (3rd part of the *ODD element*). So, for our agent-based model, we should consider how we want to realize space and time as well as scale and shape in the simulation. This includes **specifying what a *spatial unit* in the model represents in reality as well as what a *time step* means in reality** [1].

Processes of the Agent-Based Model. The third *ODD element* summarizes **what happens in the model and in what order**. So, when creating our agent-based model, we should consider the time step and the sequence in which *individual processes* (see Table 1) occur. To monitor this, we should create a sequence of *actions*, for each action, we should decide *how and which entity(ies)* are needed, and *which state variables change* as a result, as well as the *order* in which the entities exhibit their behavior [1].

Design Aspects. The fourth *ODD element* looks at *design concepts*. Some of these are also relevant for our approach to design agent-based models. The concept *fundamental principles* includes reference to already established theories, idea, hypotheses, and prior modeling approaches. For the development of our agent-based model, we check **whether the model considers an idea that has already been addressed in theory** and decide whether we want to use a theory for agent behavior.

As shown earlier, *emergence* is a fundamental feature of agent-based models. We should also consider which mechanisms predictably lead to which outcomes as well as **which outcomes are not predictable** (***emergent***) **for our model**.

The concept *adaptation* specifies the adaptive behavior of agents. For our model, we should also describe each stimuli and its triggered responses in **agents**, i.e., **when they change their behavior and to what extent adaptively**. We can decide whether agents choose a behavior that directly serves their goal (*direct goal seeking*) and therefore choose a measure to (probabilistically) weigh which behavior is most likely to achieve the goal.

According to the concept *learning*, agents gain experience and change how they adaptively adjust their behavior over time based on their experience. Here, learning does not mean that state variables cause agents to adapt their behavior, but that **agents change the methods of how they make decisions** (e.g., algorithms). For our model, we should define whether there is a form of learning and how it should be implemented if so.

According to the concept *prediction*, we can **specify whether and how agents predict future conditions and decision consequences**. To do this, we need to consider what internal models of future conditions and decision consequences our agents should have to make predictions for decision making. There are *explicit* and *implicit predictions*.

The concept of *sensing* considers **what agents know and what information they possess**. Knowledge determines how agents behave. For our model, we should specify what information (limited/only local) agents possess. We have to decide how an agent perceives the state variables of which entities. Similarly, we need to define which entities (e.g., spatial unit they reside on) they collect values from. Agents can solicit values not only locally, but also across networks and globally.

Agents can *interact* (*concept*) with each other globally, but also locally. The agents can interact *directly* or in a *mediated* manner. If an agent *directly interacts* with other agents it is a *direct interaction*. In contrast, in *mediated interaction*, agents influence each other only indirectly, for example, when they produce a common resource. *Interaction* also includes communication, which is the exchange of information in simulations. For our model, we need to specify which agents interact with each other and how [1].

The concept *stochastic* describes which processes are determined by pseudo-random numbers and how. ***Stochastic processes* are useful when we want to achieve variation in the model**, but do not want to specify which mechanisms cause the variability and how. *Stochastics* is often used for randomly creating state variables at the beginning of agent-based models. In addition, random numbers are used to shape the behavior of agents so that they exhibit different behaviors with the same frequency as observed in real people. For our model, we should also decide, whether pseudorandom numbers should be implemented and for which.

Collectives represent an intermediate level in the organization of agent-based models. They are **aggregations of agents that influence agents and are**

influenced by agents (e.g., social groups, schools of fish). *Collectives* can either be implemented as an *emergent* property of agents (e.g., flock of birds) and not explicitly represented in the model, or *explicitly* defined as an entity type (with shared state variables and behaviors) (e.g., dog packs, political parties).

Observation considers **how information from the agent-based model is collected and analyzed**. Thus, for our model, we should also consider how we collect the information from the agent-based model, or rather, which outcomes we observe (since not all outcomes can be observed) to analyze them later.

Initialization. At one point we should consider **what we need to set up the model** (see *ODD concept 5 Initiation*). We should consider which processes or sub-models we will implement only at the beginning of the simulation. To do this, we should assign numbers to entities, consider what locations the agents are in at the beginning, how the agents are networked together, and what collectives exist at the beginning. In addition, we should consider whether we want to simulate only a specific case or study system or whether it should be more generic and applicable to different sites. Further, we can **decide if our model is always initialized the same way or if we want to generate different scenarios with different initialization** (like different number of agents). So, we should consider whether we are interested in the results regarding different initial states or the results due to the change of aspects during the model. We also need to consider what data we want to use to build our model (for example, to initialize the agent populations). We should justify why we want to use which initialization methods. For example, we should explain which state variables vary between entities and in what way. We can also consider whether it makes sense to run different simulation experiments to visualize the effects of different initialization assumptions [1].

Model Dynamics. In addition to initializing the model, we also need to consider the dynamics of the model. **The model can be dynamic in that the input data can include time series of variable values or outcomes that affect the simulation**. Frequent use is made of environmental variables that change within simulation runs. Input data are often values observed in reality and therefore have statically realistic properties. External models may also be used to generate the input. Similarly, input data can be specified by external events affecting the model during the simulation (for example, times when new agents are created) [1].

Concretization of Idea. Before we can start to actually implement the agent-based model, it is important to have a ***concrete idea* of what the agent-based model should show** respectively which question it should answer. Thus, we must reflect, why the agent-based model should be build.

Basic Elements. When we think about what aspects we need to create agent-based models, we can also think of the basic elements (see Sect. 2.2) of every agent-based model: *agents*, *environment*, *network/topology* (see Table 1).

Table 1. Requirements to design agent-based models

Requirements
1. Idea/suitable research question/purpose of model
2. Added value
3. Identify emergent aspect/Identify a kind of dynamic
4. Decide for a way to design agent-based models
5. Choose processes and adaptive key processes
6. Think of contexts well-suited for agent-based models
7. Define entities, state variables, and scales
8. Design concepts of model
9. Define initial state of agent-based model
10. Define model dynamics
11. Concretization of idea
Agents:
12. Heterogeneity/Homogeneity? Different subgroups? Appearance
13. Deterministic vs. random behavior
14. Which behavior should agents show?
15. Define (sets or subsets of) attributes (for defined state)
16. Define interactions between agents
Environment:
17. Which information is given to the agents?
18. How does it influence the actions of agents?
Topology:
19. What should it represent? Physical/geographical/social... network
20. Static vs. dynamic
21. Define number of topologies (1 or more)

Agents. To design the *agents*, we must decide, **whether and *how many different groups of agents* we have or whether each agent is different**. We can consider how the *agents should look like*: Like humans, like animals or anything else? Should all agents look the same? Further, we can decide, whether we want the agents to **behave *deterministically or randomly*** (see Sect. 2.2) and ***what the agents can do*** in the simulation. We can reflect, whether we want to have a realistic aging process with agents that are born and die. Additionally, we have a lot of options, what the agents can do in the simulation, such as learn, interact physically, exchange opinions, adapt, infect each other, change and they can have emotions, beliefs, desires, goals, intentions and plans.

When we want to design agents, we need to consider, that **some characteristics (4) of agents are essential, whereas others are optional**. Wooldridge and Jennings [41] collected characteristics common to most agents and other studies explained the characteristics further [11,15,24,42]: Agents always have a

boundary (*self-contained*). They behave independently and gain information by interactions with other agents and the environment (*autonomous*) [24]. Based on the information, they make independent decisions. They can interact with other agents in at least a certain range of situations. This interaction does not necessarily affect their autonomy. Thus, agents are also active rather than purely passive [7]. In addition, agents have a *defined state* consisting of sets or subsets of attributes. All agent states combined with the state of the environment form the system state. Further, agents interact with other agents (*social*) [24].

Besides, agents can show optional characteristics. In the simulation, agents may modify their behavior according to rules (*adaptive*). Further, they probably adjust their behavior to reach a goal (*goal-directed*). Additionally, agents can be diverse in their attributes and their behavior (*heterogeneous*)[1] [24]. Normally autonomous individuals evolve. If there are groups of agents, they have usually formed from the bottom up by similar autonomous individuals joining together [7].

That agents (can) exert influence independently in a model makes them *active*. Agents can be active in different ways: They can be *proactive/goal-oriented* and seek to achieve goals through their behavior. Agents can also be *reactive/perceptive* and are aware of or have a sense of their environment in the simulation. They may have prior knowledge, such as a mental map of their environment, which increases their awareness of other entities, obstacles, or desired targets in the environment. Agents can also be active by being *interactive/communicative*. They can communicate with other agents within their neighborhood or environment in the simulation and, for example, request certain attributes. They may ignore input that contradicts a desired threshold. In addition, agents are *mobile*. They can move around the environment of the model. Finally, agents can also be able to learn or adapt their behavior adaptively and then have a kind of memory [7].

Environment. **Agents are located in the environment and operate within the space defined by the environment** [7]. The environment also **provides the topology and supports the interaction of agents with each other and the environment**. Thus, to create the environment, we must know how the agents are connected to each other. Additionally, we can decide, *which information the environment provides to the agents* and *how the environment influences the actions of the agents* [24].

Topology. The topology indicates, **how agents are connected to each other**[2]. We can decide, whether the topology should represent a ***physical/geographical network or e.g., a social network*** and whether it is ***static or dynamic***. Further, we should consider, whether our model includes only *one or different*

[1] Examples of an agent-based simulation with homogeneous are the well-known *forest fire* and *schelling* [33] model.

[2] Examples for topology include a 2-dimensional grid, network topology or geographic information system topology.

topologies. For example, agents can have different neighborhoods (e.g. geographical, social, ...). Usually only local information is available to the agents. This is typically information from an agent's neighbors [24].

3.3 Suitability of Modelling Languages for Designing Agent-Based Models

We know that modelling languages have been used to design agent-based or other simulations. We understand the specifics of each type of modelling languages from Sect. 2.4. In this section, we discuss the suitability of specific modelling languages (i*Star, UML) and why they can be used to design agent-based models.

i*Star Modelling Language. With agent-oriented modelling becoming popular, several modelling languages have been proposed for the construction of agent-oriented models. **i*Star**, being a goal modelling language **allows modellers to define goals, actors and roles** in the model as well as **dependencies between them in a clear way**. It proposes use of two models: *Strategic Dependency (SD)* which represents the actors as *nodes* and their dependencies as *relationships*. The connected actors will have a common objective, which is represented as *intentional elements*. The intentional elements can be *resources, goal, or softgoals*. The second model is the *Strategic Rationale (SR)* model, which links the relationships defined in SD model into the boundary of the actor and refines the SD model with reasoning. Elements in the SR model are linked in two ways: *Means-end links*, where *mean* are one or more intentional elements that contribute to an end—which can be goal, task, resource or softgoal. *Task decomposition*, which relates to the decomposition of a task into different intentional elements [3]. To get an intuitive overview of the environment and the actors being modelled, the graphical notations provided by the language can be used [34]. The literature points out that i*Star language does not have a language definition and that this was intentional as it gives the language flexibility. But, this flexibility also has given rise to ambiguity while using the language notations [3].

Unified Modelling Language (UML). As the name suggests, the Unified Modelling Language was defined as a *group* or *union* of modelling languages, where several modelling languages are combined each being able to model a specific aspect of a system. Unified Modelling Language is the most popular language for modelling object oriented systems and has been standardized with the main purpose of having an agreement on the commonly accepted notation and abstract syntax for all the diagram types [10,23]. UML was not intended to be a new language but a language that includes all best approaches of available modelling languages when it was introduced. UML follows the traditionally distinguished aspects of the system and provides sub-languages to model *structural* and *behavioural* parts of the system. It provides *class* and *object* diagrams which are derived from the *Entity-Relationship Diagrams* to model structural system

aspects. Objects are described by attributes and operations which might change the object's state. The structural relationships are defined by associations and constraints. UML provides several types of diagrams for modelling behavioural aspects of the system. *Use-case diagrams* help in getting the overall functionality (use cases) of a system by defining the actors and their use cases. *Activity diagrams* help to depict the control-flow of the system, *Sequence diagrams* depict the behaviour of a system in a specific scenario, *State-machine diagrams* describe the behaviour of an object over time [10].

Comparison of Languages. As we described, every language comes with its own unique properties and addresses a very specific problem in the modelling world. There are several ways in which the modelling languages are compared in the literature [14,23,26,35]. We found the **semiotic approach to suit our requirements for comparing the modelling languages** the best [23,26].

Semiotic Framework. **Semiotics focuses on the functions that are used to communicate either verbally, non-verbally, or visually**. The semiotic method fits well to our approach as we intend to use the modelling language mainly for communicating the nuances of ABM. We refer the semiotic framework used by Matulevič ius and Heymans [26] to check if the requirements we developed fits a standard framework. Although the semiotic framework has been used to compare the quality of modelling languages, in this paper **we use the framework to check the suitability of modelling languages to design agent-based models**. We then compare the requirements of the agent-based models against the requirements fulfilled by each model, to understand which modelling language approach suits best for designing agent-based models.

The semiotic quality *(SEQUAL) framework* represents a constructivistic world-view, where model creation is seen as a part of communication between the team or users knowledge about the domain, which changes as the modelling takes place. **The framework divides the language quality into five areas:** *Domain appropriateness* that relates the language to the domain, which means that there are no domain statements that cannot be expressed using the language. *Participant knowledge appropriateness* that adheres to the knowledge the participant has about the language which need not be static. *Knowledge externalizability appropriateness* which takes care of how relevant knowledge of the participant can be expressed in the language. *Comprehensibility appropriateness* relates to the understanding of all the statements in the language by the language user. *Technical actor interpretation appropriateness* relates to the formal language requirement by the technical actor (tools).

4 Evaluation of Modelling Languages

As the languages we compare are very different from each other, we use the principles defined by *SEQUAL framework* as a structure for a common ground for mapping our requirements and evaluating other modelling languages.

Sequel Framework	Criteria	ABM Requirements	i* language	UML
Domain Appropriateness	What are the views covered by language? (structural, functional, behavioural, rule-based, actor, role)	Identify the emergent aspect	SD and SR diagrams	NA
		Identify a kind of dynamic	SD and SR diagrams	NA
		Deterministic vs random behaviour?	Not well defined	Activity diagrams,
		Which behaviour must agent show?	Not well defined	Use-case, State-machine diagrams
Domain Appropriateness	Requirement definition to model the ABM	Concretization of the Idea	SD diagrams	Class diagrams
		Decide for a way to design ABM	SD diagrams	Use-case / Class diagrams
		Additional empirical Analysis?	Intentional elements	NA
		Additional verification against real data?	SD Diagrams	NA
		Theoretical assumptions?	SD Diagrams	NA
		Which behaviour should agents show?	SD and SR diagrams	Sequence diagram
		Which information is given to the agents?	NA	Class diagrams
		Define model dynamics	SD and SR diagrams	Structural and Behavioural diagrams
		Define entities, state variables and scales	NA	Class, State-machine diagrams
Comprehensibility Appropriateness	Does it support graphical representation? - to help in communication	Representation of agent, topology and environment	Agent - actor and dependencies, topology and environments are NA	Structural diagrams for topology and environment and Behavioural diagrams for agent
		How does the information influence the agent's action in an environment?	Means-end links, Task decomposition	Activity diagrams
		What does the topology represent? Physical, geographical, social network	NA	Class diagrams
		Define number of topologies	NA	Class diagrams
		Define interaction between agents	Mean-end links	Use-case diagrams
		Define sets or subsets of attributes of agents	NA	Object diagrams
		Representation of different agents - homogeneity/heterogeneity, subgroups	NA	Object diagrams
		Define initial state of ABM	NA	State-machine diagram
		Design model concepts	SD diagrams	Structural diagrams
		Define entities, state variables and scales	Entities - actors, State variable and scales - NA	Object and Class diagrams
Technical Actor Interpretation Appropriateness	Formal semantics? — ensure that the model is not misunderstood	Define entities, state variables and scales	Entities - actors, State variable and scales - NA	Object and Class diagrams
		Representation of agent, topology and environment	Not so well defined	Well defined semantics
		Define interaction between agents	Possible ambiguities in definition	Well defined semantics in class and object diagrams
		Define sets or subsets of attributes of agents	Possible ambiguities in definition	Well defined semantics in class and object diagrams
Comprehensibility, Knowledge externalisability appropriateness	Well defined constructs	Representation of agent, topology and environment	NA	NA
		Define entities, state variables and scales	NA	Well defined constructs
Comprehensibility, Domain, Knowledge externalisability appropriateness	Expressivenes of power— relation between construct and views	Representation of agent, topology and environment	Only 1 view available - SD	One diagram to give an overview of the system - classdiagram
		Define interaction between agents	SD and SR diagrams, SR being detailed	More than one available to give differed perspectives of interaction

Fig. 2. Evaluation of *requirements to design agent-based models* and how the modelling languages meet these requirements

4.1 Requirements to Design Agent-Based Models and Solutions Offered by Modelling Languages

Although *SEQUAL* provides fundamental principles it is still abstract. An additional mapping to the principles which is detailed enough to evaluate the modelling languages is required. We adapted the mapped categories used by Matulevičius and Heymans [26], and compared i*Star and UML against only those categories that fit our *requirements for agent-based models* (see Fig. 2). Firstly, we found that the requirements that we have constructed for agent-based models satisfy four out of five principles of the *sequel framework*. The principle that we did not address is the *Participant knowledge appropriateness*, which adheres to our knowledge of using agent-based models, which as mentioned in the paper has improved with the usage. For the first two agent-based model requirements: *Identify the emergent aspect* and *a kind of dynamic*, we did not find a suitable diagram in both the languages, that fulfills the requirements. We added *NA* where we did not find a suitable language construct to meet the requirements.

4.2 Does the Chosen Modelling Language Offer a Real Benefit as a Basis for an Agent-Based Model?

Both modelling languages (i*Star and UML) offer certain benefits and fail at certain places in fulfilling the model requirements identified for agent-based models. Although i*Star is developed for designing agent-based models, the behavioural view is less defined in the language, which is of concern as we require more well defined behavioural views for designing agent-based models. But, the i*Star language offers language constructs that are focused on goals, which also an important feature to have in mind while designing ABMs. On the other side, UML offers more than five different sub-languages for designing behaviour. Having more than one diagrams for designing a specific part of ABM would help in having more than one perspective of that aspect—which is also an important feature to have when designing ABMs. Secondly, i*Star does not offer well defined guidelines and methodologies, while UML has been standardised by the Object Management Group. Hence, the language ambiguities are taken care of.

From Fig. 2 **we discern that UML is more suitable for designing agent-based models** as it fits to be a more complete language offering different types of diagrams or sub-languages to design different parts of the agent-based model or different perspectives of an agent in the agent-based model. Although the i*Star language has its ambiguities, the i*Star language is more suitable for designing agent-based models when we want to know outcomes of the model like identifying the emergent aspects or testing if additional empirical data is required.

5 Conclusion

With this paper we evaluated different ways to design agent-based models. **We identified some leverages of using modeling languages to design agent-based models**. In contrast, we found that it takes longer time to design an agent-based model using a modelling language compared to designing it without. Still, when designing agent-based models, modellers should in general spend time in reconsidering, why the agent-based model is designed and whether it is designed to answer a research question of interest.

Although, we only reflected theoretically, whether modelling languages can help in designing agent-based models, this study can be used as a motivation for conducting and empirical analysis in this regard. Besides, we only focused on two modelling-languages, that we identified to be promising for creating agent-based models. **In the future, we would like to test our suggested approach to actually create agent-based models**. We plan to create agent-based models combining the use of the *ODD protocol* and the use of each modelling language we looked at in this study. In this way, we can evaluate better, whether it makes sense to include a modelling language in the design of agent-based models.

References

1. Grimm, V., et al.: "Supplementary file s1 to: Grimm, V., et al. (2020) 'the odd protocol for describing agent-based and other simulation models: a second update to improve clarity, replication, and structural realism". J. Artif. Soc. Soc. Simul. **23**(2) (2020)
2. Amouroux, E., Gaudou, B., Desvaux, S., Drogoul, A.: ODD: a promising but incomplete formalism for individual-based model specification. In: 2010 IEEE RIVF International Conference on Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF), Hanoi, pp. 1–4 (2010)
3. Martínez, C.P.A., et al.: A comparative analisys of i*-based agent-oriented modeling languages. In: Proceedings of the SEKE 2005, the 17th International Conference on Software Engineering & Knowledge Engineering: Technical Program, 14–16 July 2005, Taipei, Taiwan, Republic of China, pp. 43–50 (2005)
4. Bonabeau, E.: Agent-based modeling: methods and techniques for simulating human systems. Proc. Natl. Acad. Sci. USA **99**(Suppl 3), 7280–7287 (2002). https://doi.org/10.1073/pnas.082080899
5. Bruch, E., Atwell, J.: Agent-based models in empirical social research. Sociol. Methods Res. **44**(2), 186–221 (2015). https://doi.org/10.1177/0049124113506405. PMID: 25983351
6. Byrne, D.S.: Complexity Theory and the Social Sciences: An Introduction. Business and the World Economy. Routledge, London (1998). ISBN 9780415162968. https://books.google.de/books?id=NaVSZXVdc-0C
7. Castle, C.J., Crooks, A.T.: Principles and concepts of agent-based modelling for developing geospatial simulations (2006)
8. Conte, R., et al.: Manifesto of computational social science. Eur. Phys. J. Special Topics **2014**(1), 325–346 (2012). http://wrap.warwick.ac.uk/67839/
9. Curtis, B., Kellner, M.I., Over, J.: Process modeling. Commun. ACM **35**(9), 75–90 (1992)

10. Engels, G., Heckel, R., Sauer, S.: UML—a universal modeling language? In: Nielsen, M., Simpson, D. (eds.) ICATPN 2000. LNCS, vol. 1825, pp. 24–38. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44988-4_3
11. Epstein, J.M.: Agent-based computational models and generative social science. Complexity **4**(5), 41–60 (1999). https://doi.org/10.1002/(SICI)1099-0526(199905/06)4:5⟨41::AID-CPLX9⟩3.0.CO;2-F(199905/06)4:5⟨41::AID-CPLX9⟩3.0.CO;2-F. ISSN 1076–2787
12. Epstein, J.M.: Generative Social Science: Studies in Agent-Based Computational Modeling (Princeton Studies in Complexity). Princeton University Press, Princeton (2007)
13. Fanelli, D.: Opinion: Is science really facing a reproducibility crisis, and do we need it to? Proc. Natl. Acad. Sci. USA **11**, 2628–2631 (2018)
14. Felderer, M., Herrmann, A.: Comprehensibility of system models during test design: a controlled experiment comparing UML activity diagrams and state machines. Softw. Qual. J. **27**(1), 125–147 (2019)
15. Franklin, S., Graesser, A.: Is it an agent, or just a program?: a taxonomy for autonomous agents. In: Müller, J.P., Wooldridge, M.J., Jennings, N.R. (eds.) ATAL 1996. LNCS, vol. 1193, pp. 21–35. Springer, Heidelberg (1997). https://doi.org/10.1007/BFb0013570
16. Grimm, V., et al.: A standard protocol for describing individual-based and agent-based models. Ecol. Model. **198**(1–2), 115–126 (2006). https://doi.org/10.1016/j.ecolmodel.2006.04.023. ISSN 0304–3800
17. Grimm, V., et al.: The odd protocol for describing agent-based and other simulation models: a second update to improve clarity, replication, and structural realism. J. Artif. Soc. Soc. Simul. **23**(2), 7 (2020). https://doi.org/10.18564/jasss.4259. ISSN 1460–7425. http://jasss.soc.surrey.ac.uk/23/2/7.html
18. Grimm, V., Berger, U., DeAngelis, D.L., Gary Polhill, J., Giske, J., Railsback, S.F.: The odd protocol: a review and first update. Ecol. Model. **221**(23), 2760–2768 (2010). https://doi.org/10.1016/j.ecolmodel.2010.08.019. ISSN 0304–3800. https://www.sciencedirect.com/science/article/pii/S030438001000414X
19. Hahn., C., A domain specific modeling language for multiagent systems. In: Proceedings of the 7th International Joint Conference on A"Utonomous Agents and Multiagent Systems, vol. 1, pp. 233–240. Citeseer (2008)
20. Helbing, D., Balietti, S.: How to do agent-based simulations in the future: from modeling social mechanisms to emergent phenomena and interactive systems design why develop and use agent-based models? Int. J. Res. Market., 1–55 (2011). https://doi.org/10.1007/978-3-642-24004-1. http://www.santafe.edu/media/workingpapers/11-06-024.pdf
21. Humphrey, W.S., Kellner, M.I.: Software process modeling: principles of entity process models. In: Proceedings of the 11th International Conference on Software Engineering, pp. 331–342 (1989)
22. Kavakli, E.: Goal-oriented requirements engineering: a unifying framework. Requirements Eng. **6**(4), 237–251 (2002)
23. Krogstie, J., Using a semiotic framework to evaluate UML for the development of models of high quality. In: Unified Modeling Language: Systems Analysis, Design and Development Issues, pp. 89–106. IGI Global (2001)
24. Macal, C.M., North, M.J.: Tutorial on agent-based modeling and simulation. In: Proceedings of the 2005 Winter Simulation Conference, pp. 2–15 (2005)
25. Matulevicius, R.: Improving the syntax and semantics of goal modelling languages. In: iStar, pp. 75–78 (2008)

26. Matulevičius, R., Heymans, P.: Comparing goal modelling languages: an experiment. In: Sawyer, P., Paech, B., Heymans, P. (eds.) REFSQ 2007. LNCS, vol. 4542, pp. 18–32. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-73031-6_2
27. Monks, T., Currie, C.S., Onggo, B.S., Robinson, S., Kunc, M., Taylor, S.J.E.: Strengthening the reporting of empirical simulation studies: introducing the stress guidelines. J. Simul. **1**(13), 55–67 (2019)
28. Peng, C., Guiot, J., Wu, H., Jiang, H., Luo, Y.: Integrating models with data in ecology and palaeoecology: advances towards a model-data fusion approach". Ecol. Lett. **14**(5), 522–536 (2011)
29. Polhill, J.G., Parker, D.C., Brown, D., Grimm, V.: Using the ODD protocol for describing three agent-based social simulation models of land-use change. J. Artif. Soc. Soc. Simul. **11**(2), 1–3 (2008)
30. Railsback, S.F.: Concepts from complex adaptive systems as a framework for individual-basedmodelling. Ecol. Model. **139**(1), 47–62 (2001)
31. Rand, W., Rust, R.T.: Agent-based modeling in marketing: guidelines for rigor. Int. J. Res. Market. **28**(3), 181–193 (2011). https://doi.org/10.1016/j.ijresmar.2011.04.002. ISSN 01678116
32. Retzlaff, C.-O., Ziefle, M., Calero Valdez, A.: The history of of agent-based modeling in the social sciences. In: Duffy, V.G., (ed.) HCII 2021, LNCS 12777, pp. 304–319. Springer, Cham (2021)
33. Schelling, T.C.: Models of segregation. Am. Econ. Assoc. **52**(2), 604–620 (2013)
34. Yu, E.S.-K.: Modelling strategic relationships for process reengineering. Ph.D. thesis, University of Toronto (1996)
35. Söderström, E., Andersson, B., Johannesson, P., Perjons, E., Wangler, B.: Towards a framework for comparing process modelling languages. In: Pidduck, A.B., Ozsu, M.T., Mylopoulos, J., Woo, C.C. (eds.) CAiSE 2002. LNCS, vol. 2348, pp. 600–611. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-47961-9_41
36. Calero Valdez, A., Ziefle, M.: Human factors in the age of algorithms. Understanding the human-in-the-loop using agent-based modeling. In: Meiselwitz, G. (ed.) SCSM 2018. LNCS, vol. 10914, pp. 357–371. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-91485-5_27
37. Vincenot, C.E.: How new concepts become universal scientific approaches: insights from citation network analysis of agent-based complex systems science. In: Proceedings of the Royal Society B: Biological Sciences, vol. 285, p. 20172360 (2018)
38. Mitchell Waldrop, M.: Complexity: The Emerging Science at the Edge of Order and Chaos. Simon & Schuster, New York (1992). ISBN 0671767895
39. Wilensky, U., Rand, W.: An Introduction to Agent-Based Modeling: Modeling Natural, Social, and Engineered Complex Systems with NetLogo. The MIT Press (2015) ISBN 9780262731898. https://books.google.de/books?id=LQrhBwAAQBAJ
40. Wooldridge, M.: An Introduction to Multiagent Systems. Wiley, Hoboken (2009)
41. Wooldridge, M., Jennings, N.R.: Intelligent agents: theory and practice. Knowl. Eng. Rev. **10**, 115–152 (1995)
42. Wooldridge, M., Jennings, N.R.: Simulating sprawl: a dynamic entity-based approach to modelling North American suburban sprawl using cellular automata and multi-agent systems. Ph.D. Thesis (2004)
43. Zamli, K.Z., Lee, P.A.: Taxonomy of process modeling languages. In: Proceedings ACS/IEEE International Conference on Computer Systems and Applications, pp. 435–437. IEEE (2001)