Creating Agent Based Models to Support Decision Making from Semi-Structured and Unstructured Data

Joseph A. Stoffa

Improbable

Arlington, VA

joe@improbable.us

ABSTRACT

Agent Based Models are increasingly becoming a favorable technique for modeling when historical data is limited, or when it is necessary to capture the effects of complex systems in granular detail. To support decision making, models must be timely and flexible; however, defining agent behaviors for modeling and simulation often requires programming knowledge or other specialized software skills, leading to a greater barrier to entry for the layperson. A more attractive solution is an automated ingest system that extracts agent behavior directly from an organization's native documentation, such as business process workflows, standard operating procedures, or tactics, techniques, and procedures. However, the challenge with this approach is processing unstructured data into reliable and realistic agent behavior. This paper illustrates how this type of ingest and transformation was accomplished for modeling the behavior of office workers using a simulated computer network. Specifically, a series of complex tasks were derived using an ontology of atomic-level actions that directly map to semi-structured descriptions found in business operation processes; tasks were assigned to agents based on frequency of occurrence within the population of workers and by role. Agents uniquely perform tasks based on their individual knowledge, goals, available resources, and environmental effects, and as such, tasks are not prescriptive and do not explicitly detail how an agent executes them; rather, tasks are descriptive and provide a general process to follow. This work resulted in a methodology that allows agent behaviors to be derived from generalized process descriptions alone yet executed in a realistic manner.

ABOUT THE AUTHORS

Joseph A. Stoffa is an applied scientist at Improbable where he specializes in implementing scientific models into large scale multi-system simulations. Joseph obtained his MSc. in Data Science from Southern Methodist University in 2017 and has over 20 years of experience in various roles working in defense intelligence, national critical infrastructure protection, and national security. His interests included progressing the applications of agent-based modeling, human behavior modeling, and socio-technical/socio-ecological systems to provide practical solutions for real-world problems.

Creating Agent Based Models to Support Decision Making from Semi-Structured and Unstructured Data

Joseph A. Stoffa

Improbable

Arlington, VA

joe@imrobable.us

INTRODUCTION

The work described in this paper originated in a project designed to demonstrate how modelling and simulation could be used by decision makers to understand the second and third order effects that computer network outages have on their organization. To adequately simulate these effects, we needed to understand how users interacted with the network in their day to day activities. Therefore, a key aspect we wanted to capture was the individual processes employees used to complete their work. One of the goals of the project was to demonstrate how realistic human behavior could be elicited from an organization's internal documentation. Additionally, we wanted to create a simulation that was approachable to the common user, did not rely on hard-coded behaviors, and allowed users to define new behaviors. Unfortunately, due to the sensitive nature of the customer, we were not able to use any of their internal documentation. To mitigate this situation and continue moving forward with the project, we created a notional, yet analogous, company with similar business processes and types of employees that could be modelled.

This paper follows the following format: first, we provide a brief description of the notional company created for this project to give the reader the necessary context for our examples. Next, we outline how we created our ontology to serve both as a conceptual model and as a component to bridge the gap between process workflows and agent behavior. Following, we describe how overall agent behavior was implemented and how the activities described in workflows were interpreted and executed as agent behavior. We then conclude with a summary of the key components that allowed us to achieve automated ingest of agent behavior directly from semi-structured data sources and discuss how this work could be further extended to include unstructured data as well.

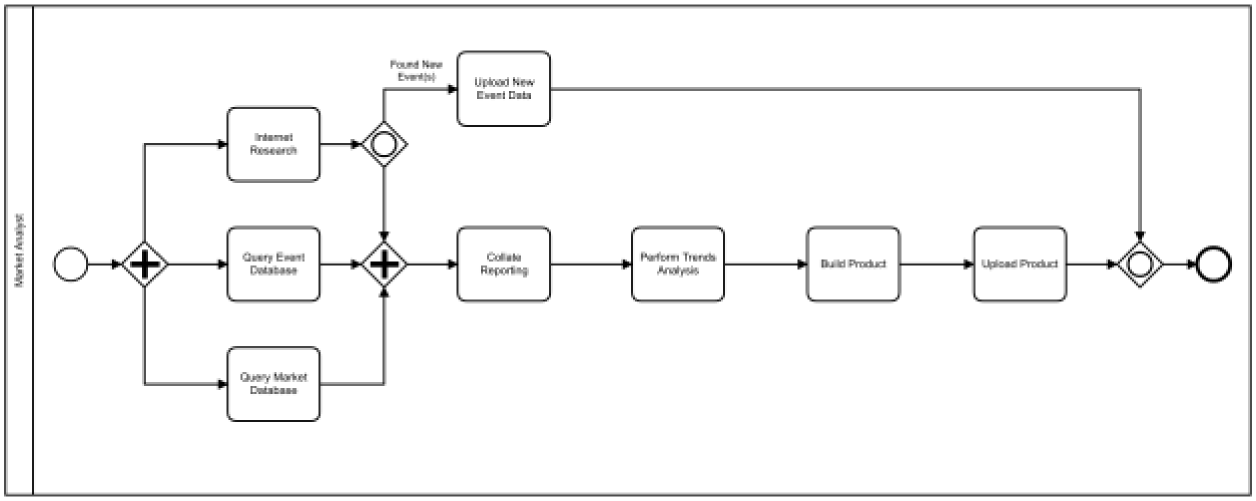
CREATING THE NOTIONAL COMPANY

To define the characteristics for the notional company, we started by devising its products and estimated the resources and personnel required to produce them. We chose to create a research and consultancy firm that specialized in analyzing the various energy sectors, since such a company has similarly skilled employees and business process as our customer. We determined the company's primary products would be reports that provide analysis of emerging trends for each sector of the energy industry, all produced by the company's analysts (GIS Analyst, Sector Analyst, and Market Researchers) who follow distinct process workflows. We consulted real-world analysts and used relevant user storyboards from prior projects to make the workflows realistic. We depicted workflows in BPMN 2.0 notation using Camunda open source software. The company's network topology was designed using industry best practices for ensuring adequate redundancy, and scaled based on the number of employees, services, and data we previously estimated. The flow of data used to build reports was generated from the distribution of references to specific types of data typically found in reporting. Since no actual reporting existed, these distributions were arbitrarily chosen. We then calculated the daily probability distributions for each data type from the distributions representing the number and types of relevant data sources that typically occurred within each report's respective reporting period. We used these daily distributions to create a data event scheduler that was later used to populate our simulated network with realistic data.

BUILDING THE ONTOLOGY

Defining an ontology is a common step in developing models for use in simulation. It has been considered as means to conceptualize models (van Dam et al 2013) and proposed as a tool to make model comparisons (Pierre et al 2010). Conceptual models are important because they provide a means for subject matter experts to communicate with the modeler. To meet our project goal of approachability, special emphasis was placed on the creation of an ontology such that once it was defined it could serve as bridge between documented process workflows and agent behavior.

We created the ontology by analyzing the organization's process workflows. Within the workflows we found a distinct set of action terms that were repeatedly used in the descriptions of activities. Terms such as *download*, *upload*, and *analyze* described "atomic" behaviors, or behaviors that could be executed by an agent directly without the need for further interpretation. We also found activities that described more complex behaviors such as *conduct internet research*, which necessitated consulting our analysts again for more explicit process descriptions. Unsurprisingly, we found that these complex activities represented subtasks that were formed from the same set of action terms found in other activities. For example, *internet research* found in Figure 1. was subsequently resolved to a subtask



that included the actions: *download*, *scan*, and *read*.

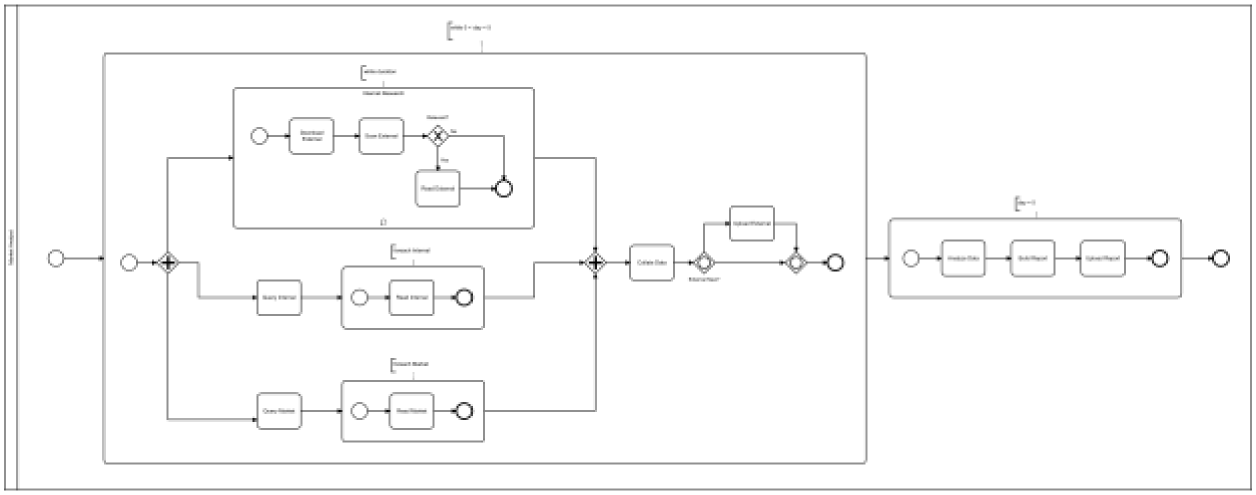


Figure 1. Initial Version of Weekly Sector Market Summary Report Workflow
Figure 2. Expanded Version of Weekly Sector Market Summary Report Workflow

Another product of the analysis of workflows was a set of common targets of actions, or the set of data types which, through the course of a workflow, were in some way processed or transferred across the network. The ontology was therefore composed of atomic actions and general datatypes with relationships that described meaningful pairings of the two. Actions were categorized in two sets: online actions that require the network (e.g. *download, up-load, query*) and offline actions that do not (e.g. *read, analyze, collate*). Online actions detail how data moves across the network whereas at a minimum, offline actions describe non-use of the network and potentially some form of data transformation as well. For example, *exploit imagery* requires an analyst's attention, representing time spent not actively using the network while the analyst studies the imagery, but the activity also outputs new network data representing the geospatial information the analyst derived from the imagery. Actions were further classified as either requiring an analyst's direct involvement (active actions) and those that did not (passive actions). Making this distinction was a prerequisite to implementing context switching between activities, allowing for the emulation of an employee's tendency to multitask.

IMPLEMENTING AGENT BEHAVIOR

Agent behavior is the product of two components: process workflows that define tasks and a set of rules for how agents select tasks. Workflows are graphical representations of tasks where individual steps in a workflow are represented as rectangles that are sequentially connected by lines and arranged in the order they are executed. Gateways, represented as diamonds in the workflow, provide additional means to control the flow of activities by allowing for parallelization and conditional branching. In the context of this paper, a *task* is defined as work executed to accomplish some goal. Tasks are made up of *activities* which describe individual steps taken in order to complete a task. Activities are comprised of an *action* and some target of the action. In our use case, tasks described any work an analyst might do during their workday, from primary business processes used to build analytical products to generic processes such as *update timecard, read and respond to email*, and *stream music*.
The workflows for our notional organization's process were depicted in BPMN and stored as XML files. We developed an ingest system for our model and simulation that automatically extracts activity objects from XML workflows, parses text descriptions found for each activity, and maps them to the appropriate actions and targets defined in our ontology. The ingest system also correctly processes gateways as control objects, start/end nodes, and subprocesses representing subtasks. The latter is implemented as a unique type of activity that controls how and when a subtask is executed within a workflow by parsing the textual control statement found in the subprocess annotation. Activity objects are stored in memory and later used as blueprints to create specific instantiations of activities

for an agent to process during simulation runtime. Activity entities list the activities that succeed them, the prerequisite actions that must be completed before they can be executed, the action to execute, the target of the action, and have an input to receive data from predecessor activities.

Prior to simulation runtime, we specify the number and types of agents as well as make task assignments. Tasks are assigned by agent type, a probability that a specific agent of that type will be given a task, and the task's priority. Core business processes are given a probability of 1.0 while non-essential activities are given lower probabilities to simulate the variability in behaviors between individual agents. At simulation runtime, agents select a task from their task list based on priority and begin executing the task's activities. Agents maintain queues for each activated task in their task list to keep track of their progress. When an agent has only *passive* activities in its queue, for example when an agent is waiting on a download or when network outages prevent them from completing an activity, the agent will select a new task. Agents will choose the new task based on three criteria: the estimated time to complete the shortest passive activity in its queue or the agent's estimate network outage (randomly sampled from a normal distribution of time), the estimated time to complete the new task, and the new task's priority. Agents will try to fill time as efficiently as possible within a specified threshold that allows an agent to select a task that exceeds the estimated downtime. In the case where multiple tasks meet this criterion the agent will select the task with the highest priority.

Also prior to simulation runtime, agents are given randomly sampled time estimates for each task they are assigned. Similarly, agents are provided general estimate for network outages that represents their personal belief for how long outages typically last. Time estimates for passive activities are given to the agent from the network model based on the available bandwidth at the beginning of a simulated data transfer.

IMPLEMENTING ACTIONS

As previously stated in the "Building the Ontology" section of this paper, we classified actions as either online or offline. Due to the nature of the system and processes, we modelled all activities that in some way include data. For example, online activities involve the movement of data within the network while offline activities describe how data is processed and potentially transformed to create new data. Within our simulation engine, two separate server-side workers were designated to operate the network and human behavior (agent) models, respectively. The agent worker is responsible for processing all offline activities while the network model handles all online activities. When an agent executes an activity that includes an online action it sends a request to the network worker to move data from one node on the network another. The network worker determines the most efficient path and initiates a flow to transmit the data. After the transfer is complete the network model notifies the agent, who in turn selects the next activity in its pending queue and continues. Since tasks are descriptive rather than prescriptive, in that they generally specify what needs to be done and to what, the exact network locations and specific data to be transferred is determined by the agent. To enable agents to make such determinations each agent is given its own set of knowledge that represents its understanding of available resources.

All agent knowledge is maintained by the simulation as a graph database that also includes associations between agents. Therefore, the knowledge graph stores who and what each analyst knows. The graph is generated prior to simulation runtime and uses characteristics such as assigned office location and agent experience level to determine an agent's specific set of knowledge and associations. Whenever an agent executes an online activity the agent worker queries the knowledge graph to determine all known network locations hat store data of type specified in the activity. If the query returns no results, indicating the agent does not know the location of that information, or the location is unreachable due to network outages, the agent will ask its peers for help. The agent worker simulates the agent's appeal to its peers by querying the knowledge graph for the agent's associations who have knowledge of an alternate location. The worker then calculates the probability that an agent gains this information after a given time step based on the number of knowledgeable associations made by an agent. If an agent learns of the new location, the knowledge graph is updated accordingly; if an agent does not learn of the location, a new probability is calculated at the next time interval. This system enables two phenomena to occur within the simulation. First, it enables an agent to provide the proper contextual information to be able to execute a generalized process description much like a real person would interpret it. Second, it allows for the dissemination of information within an organization to manifest as a realistic "snowball" effect of shifting network traffic.

Offline activities involve those actions that process and potentially transform data. They represent how a person examines data, and through some process creates new data. At a minimum, the simulation will determine the time it takes an agent to execute an action given specific input. If the action produces output data it will also calculate its size, given the type and size of the processed input data. To simulate this process, we made a simplifying assumption that a linear relationship exists between the size of data being processed and both the time needed to execute the action and the size of the output.

For each action there is a defined processing rate, and for each relevant action/datatype pair there is a conversion factor. The amount of time it takes an agent to execute an action is determined by the product of the size of data being processed (its input), its conversion factor, and the action's processing rate. The conversion factor is used to convert the size of the data into a meaningful medium for which a known processing rate exits. For example, to determine the amount of time it takes an agent to read a text document we first convert the number of bytes into number of words before applying the processing rate (words per minute). For actions that produce output data, the size of the output is determined by taking the product of the processing factor and the size of the input. Actions can take multiple types of data as input, for example *analyze* can take *GIS*, *imagery*, *internal*, and *external* datatypes as input and process them together by taking the sum of the individual products of their sizes, conversion factors, and processing rates.

Ideally, conversion factors, processing rates, and processing factors would be determined through the analysis of real-world data. For example, we would relate the average reported time it takes analysts to analyze imagery versus its size (and perhaps spatial resolution) to model their relationship. Since we are using a notional company and synthetic data, we rely on calculations made from estimates. For more common actions, such as reading and writing, we used published research to calibrate our processing rates and conversion factors. Specifically, for reading we referenced research on the average reading speeds for adults (Carver, 1992). For conversion rates we used both real world and synthetic data. Notably, for internal events (e.g. internal reporting) we used the average length of an English word (five letters) and the number of bytes per character for ASCII encoding (1 byte/char) to calculate the conversion rate of word/5 bytes. For external events (e.g. online news sources), we sampled articles from various well-known news sites such as BBC and Reuters and used simple regression to relate article size to relevant word count.

This system of granular agent actions is easily extended. Adding new actions for preexisting action types only requires specifying the factors/rates of new actions and the relevant relationship to targets. For action types that require new execution logic, we require that logic to be implemented within the simulation application.

CONCLUSION

By emphasizing agent behavior at a granular level, we created an ontology of agent actions, the targets of those actions, and the relationships between them. We classified actions into groups based on how they would be executed within our simulation and specified two separate server-side workers to handle each class. This process allowed us to map each action in our ontology to the appropriate simulation logic. Behaviors in the form of complex tasks could then be created by combining actions to depict new process as workflows. We enabled agents within our simulation to correctly process activities from generalized descriptions found in workflows by providing them with their own individual knowledge of resources and mapped their associations with other agents to support discovery.

While the methodology we described in this paper was suitable for our project, we make no claim that it can be universally applied. We believe this approach is most appropriate in domains where human behavior can be best described as distinct processes such as those typically found in industries such as manufacturing or military operations. However, we submit that placing an emphasis on defining agent behavior at the most atomic level has the potential to allow agent-based modeling to benefit from the advances made in other fields using granular ontologies.

Ontologies have been successfully used in natural language processing to extract information from unstructured text. The automated extraction of providence data from biomedical literature was achieved using well-established

granular ontologies (Valdez et al, 2016). By extension we believe that our approach could be expanded to support the ingest of unstructured textual descriptions of processes directly into agent behavior. Other potential future work could include using metaheuristic search techniques to create workflows from a set of actions given specific agent goals and environmental constraints. This future work follows how Abdulkareem et al used the metaheuris-
tic tabu search to devise a way to constructed Bayesian Networks from disconnected nodes representing individual steps in an agent decision making progress, effectively fitting agent behavior to survey data.

REFERENCES

Abdulkareem SA, Mustafa YT, Augustijn E-W, Filatova T. Bayesian networks for spatial learning: a workflow on using limited survey data for intelligent learning in spatial agent-based models. Geoinformatica. 2019; 23: 243–268. https://doi.org/10.1007/s10707-019-00347-0

Carver, R. (1992). Reading Rate: Theory, Research, and Practical Implications. *Journal of Reading, 36*(2), 84-95. Retrieved January 29, 2020, from www.jstor.org/stable/40016440

Livet, P., Miller, J.-P., Phan, D., & Sanders, L. (2010). Ontology, a Mediator for Agent-Based Modeling in Social Science. Journal of Artificial Societies and Social Simulation, 13(1), 3. http://jasss.soc.surrey.ac.uk/13/1/3.html.

Valdez, J., Rueschman, M., Kim, M., Redline, S., & Sahoo, S. S. (2016). An Ontology-Enabled Natural Language Processing Pipeline for Provenance Metadata Extraction from Biomedical Text (Short Paper). *On the move to meaningful Internet systems ... : CoopIS, DOA, and ODBASE : Confederated International Conferences, CoopIS, DOA, and ODBASE ... proceedings. OTM Confederated International Conferences, 10033*, 699–708. https://doi.org/10.1007/978-3-319-48472-3_43

van Dam, K. H., Nikolic, I., & Lukszo, Z. (2013). *Agent-based modelling of socio-technical systems* (Vol. 9). Dordrecht: Springer. doi: 10.1007/978-94-007-4933-7