INCOSE
GLRC 2018

2018 Annual INCOSE
Great Lakes Regional Conference
SYSTEMS AT THE CROSSROADS
17 - 20 October 2018 | Indianapolis, Indiana

Simulation Modeling Workshop
19 October 2018

Rainer Dronzek

- Bio
  - Rainer has held engineering and management positions in the aerospace, oil & gas and consulting industries. He began his systems engineering career on the Space Shuttle Program at the Kennedy Space Center, where he was first introduced to the art and science of simulation modeling. He has managed projects and standing teams, founded a simulation consulting firm, and organized events and conferences around simulation modeling
  - He is a Regional Director with The AnyLogic Company and holds a B.S. in Electrical Engineering from Bradley University
- Contact

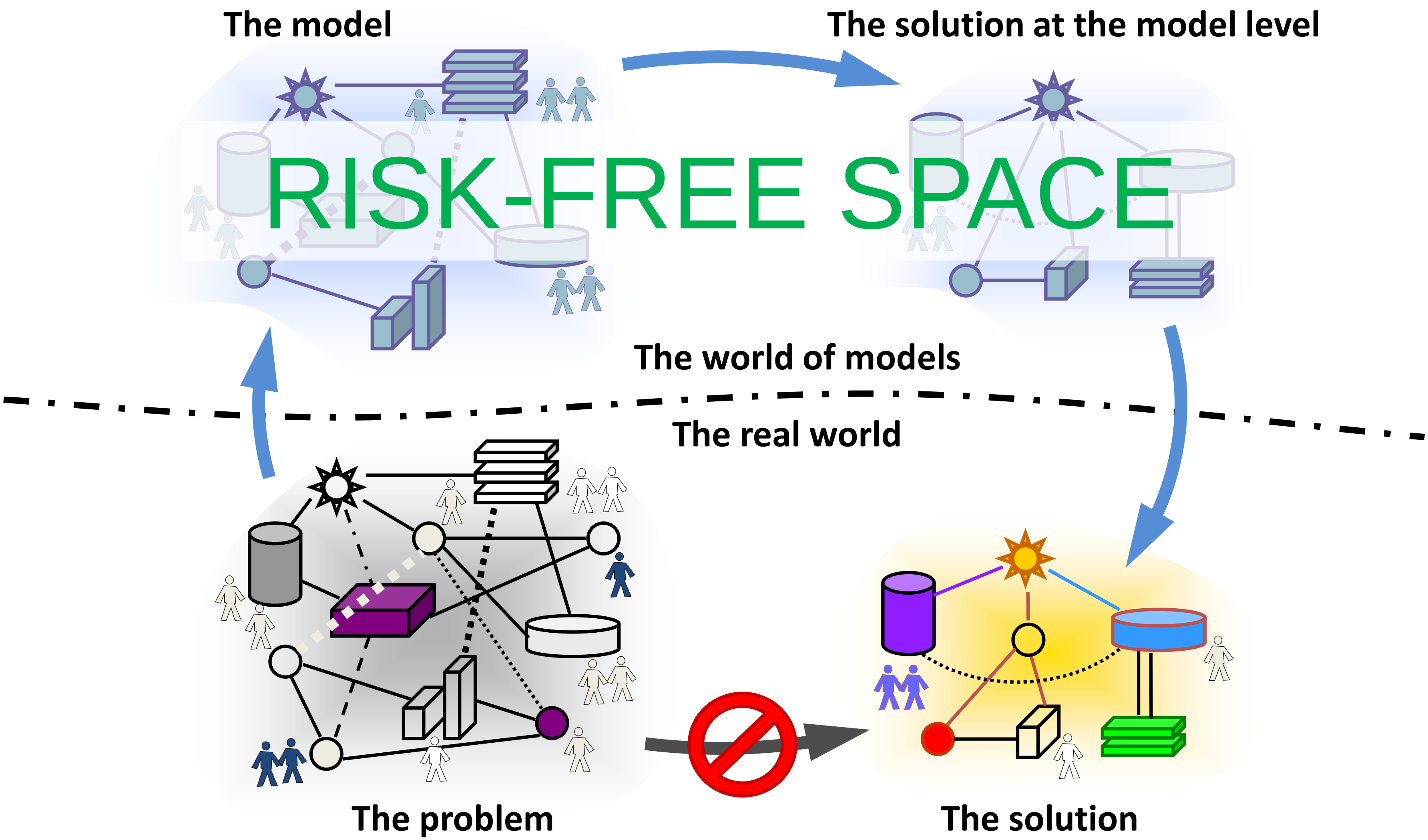
  - AnyLogic North America, 1 Tower Lane, Suite 2655, Oakbrook Terrace, IL 60181
  - Direct: 312.635.3346, Cell: 630.995.1801
  - rdronzek@anylogic.com

Agenda

- Multi-Method Simulation Modeling
- The Model Development Environment
- Discrete Event Modeling
- The Factory Model
- System Dynamics Modeling
- The Bass Diffusion Model
- Agent Based Modeling
- Disease Spread Model
- Supply Chain Model
- Railway Station Model
- Group Exercise
- Further Information
- Distribution of Simulation Software

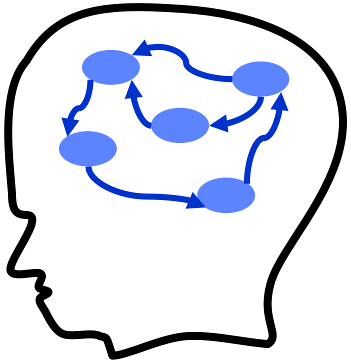
Multi-Method Simulation Modeling

Modeling

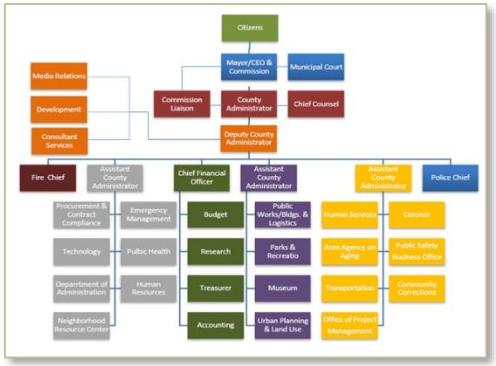

The model
The solution at the model level
RISK-FREE SPACE
The world of models
The real world
The problem
The solution

Types of models

Mental models

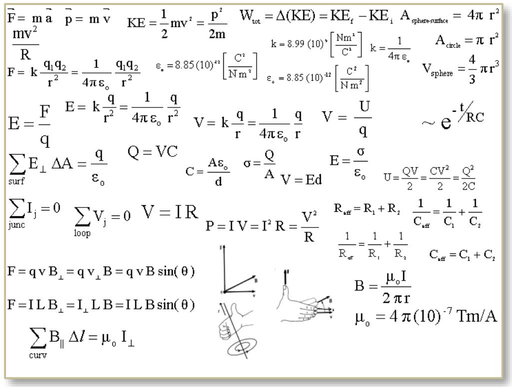


Boxes connected with lines

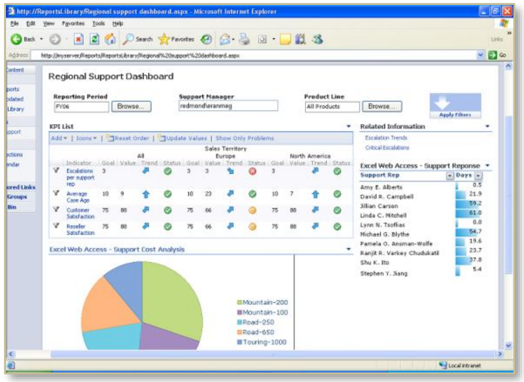


Physical models

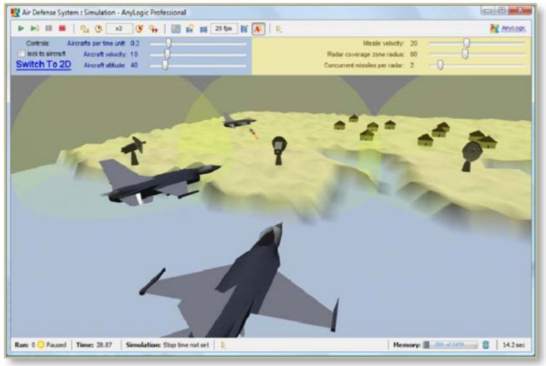
Formulas on a sheet of paper



Excel spreadsheets



Simulation models



The most popular modeling tool is:

MS Excel

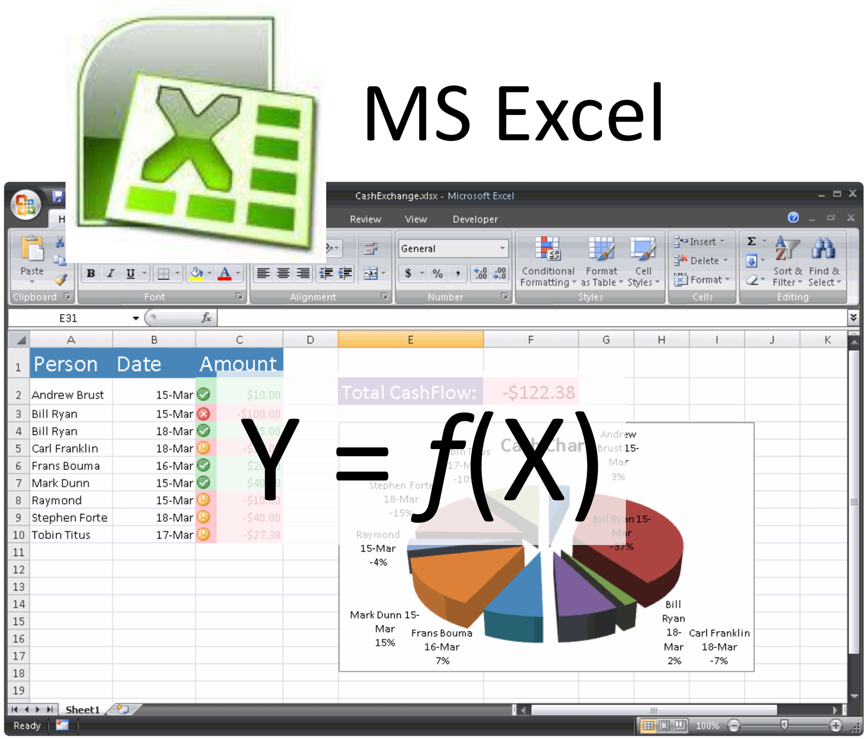
Input

X_1

X_2

X_3

X_4



$$Y = f(X)$$

Output

Y_1

Y_2

Y_3

Y_4

Analytical solution

(formulas and scripts)

However…

- You can find an analytical solution if:
  - The number of parameters is 'manageable'
  - Behavior is linear
  - Dependencies are clear, easy to build a mental model

- But what if:
  - Too many parameters
  - Non-linear, non-obvious influences
  - Time and causal dependencies
  - Counter-intuitive behavior
  - Uncertainty (stochastic system)

Example: Bank

- A simplistic case:
  - On average 10 clients per hour
  - Only one teller at the counter
  - Mean service time is 5 minutes
- We want to find out:
  - Mean waiting time in the queue
  - [Other metrics can be derived from that one]

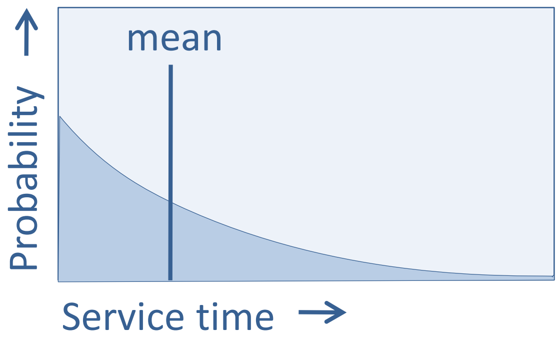
- It'll take you a few seconds to find the analytical solution:

Mean waiting time* $w = \dfrac{\lambda b^2}{1-\lambda b}$, where λ - arrival rate, b - mean service time

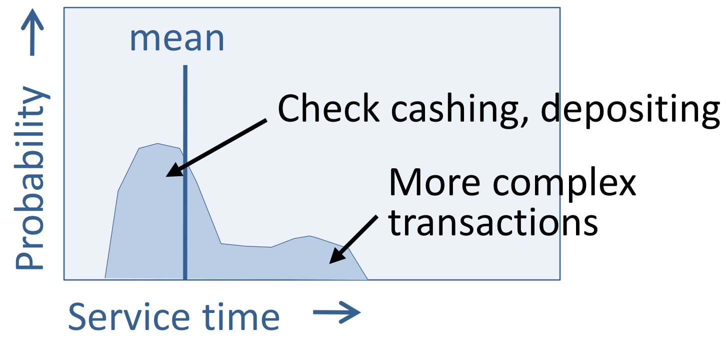
* This holds only for a Poisson stream of clients (independent arrivals with constant rate) and exponentially distributed service time.

Bank. Assumptions of the analytic approach

- What do these assumptions mean?
  - Independent arrivals of clients – this should be an OK assumption for the bank
  - Exponentially distributed service time:



This is far from reality. The distribution is more likely of this shape:



- Then the Internet search will suggest another formula:

$$w = \frac{\lambda b^2 (1 + C_b^2)}{2(1 - \lambda b)}$$, where C_b - coefficient of variation of service time

Bank. What if a bit less simplistic case?

- Let there be several (*K*) tellers
  - This is so-called "multi-server queue model". The analytic solution*:

$$w = \frac{Pb}{K(1-\rho)}$$, where $\rho = \frac{\lambda b}{K}$ - system utilization,

$$P = \frac{(K\rho)^K}{K!(1-\rho)} P_0$$, where $$P_0 = \left[\frac{(K\rho)^K}{K!(1-\rho)} + \sum_{i=0}^{K-1} \frac{(K\rho)^i}{i!} \right]^{-1}$$

- probability of all tellers being busy

- probability of "no clients in the bank"

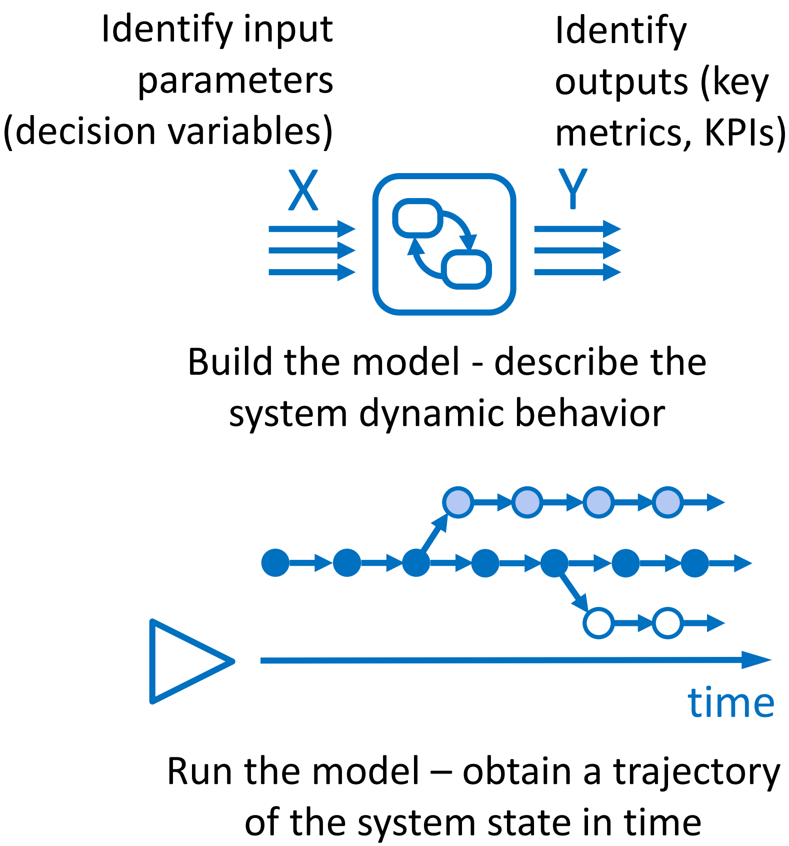
* This, however, is valid only for Poisson stream of clients and exponentially distributed service time.

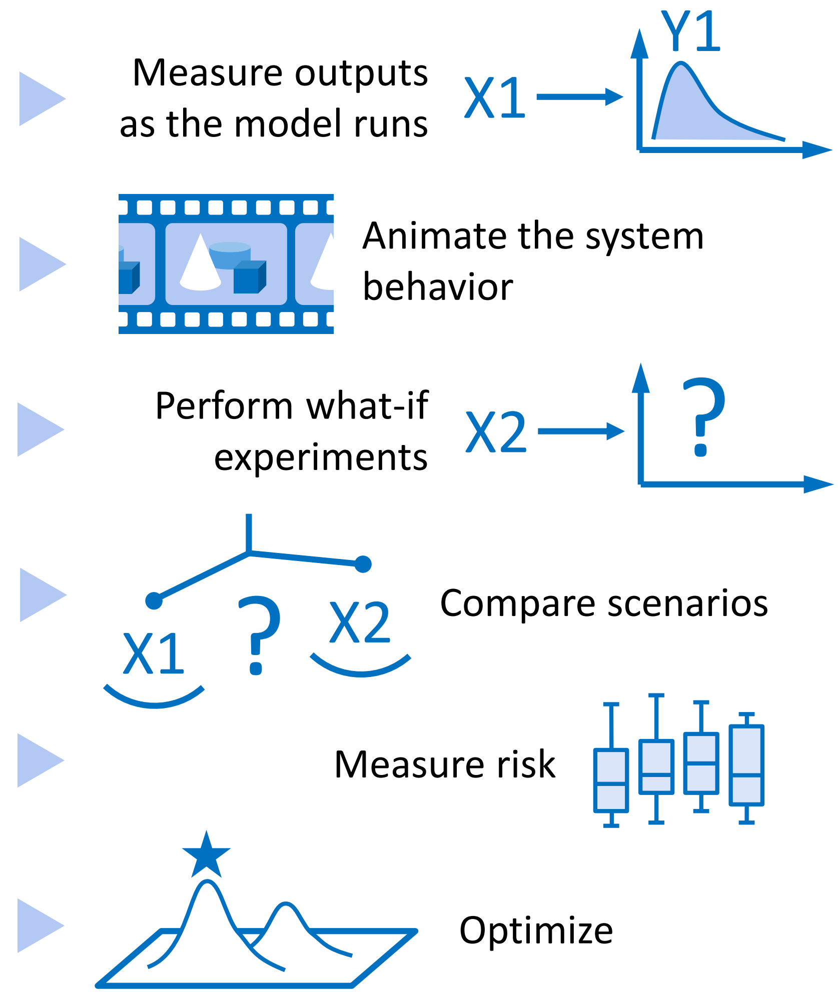
- And what if service time has a different distribution?
  - Even for such a simple system there is no analytic solution

Bank model

- In the real bank the process is far more complex:
  - Some transactions can be done only by some particular employees
  - The client can be redirected to other employees
  - The tellers may share resources, such as a printer or a copier
  - Different employees may have different skills and performance
  - Etc.
- The analytic solution probably does not exist
  - Even if it exists, who will find it for you?
  - Almost any change in the process makes the previous analytic solution void
- The only analysis method for such systems that has foreseeable complexity and guarantees the result is: simulation modeling

Simulation modeling





- Dynamic simulation enables much more detailed analysis and can solve problems that spreadsheet-based or LP-based analytics can't

Application areas

High abstraction level
[minimum details
macro level
strategic level]

Medium abstraction level
[medium details
meso level
tactical level]

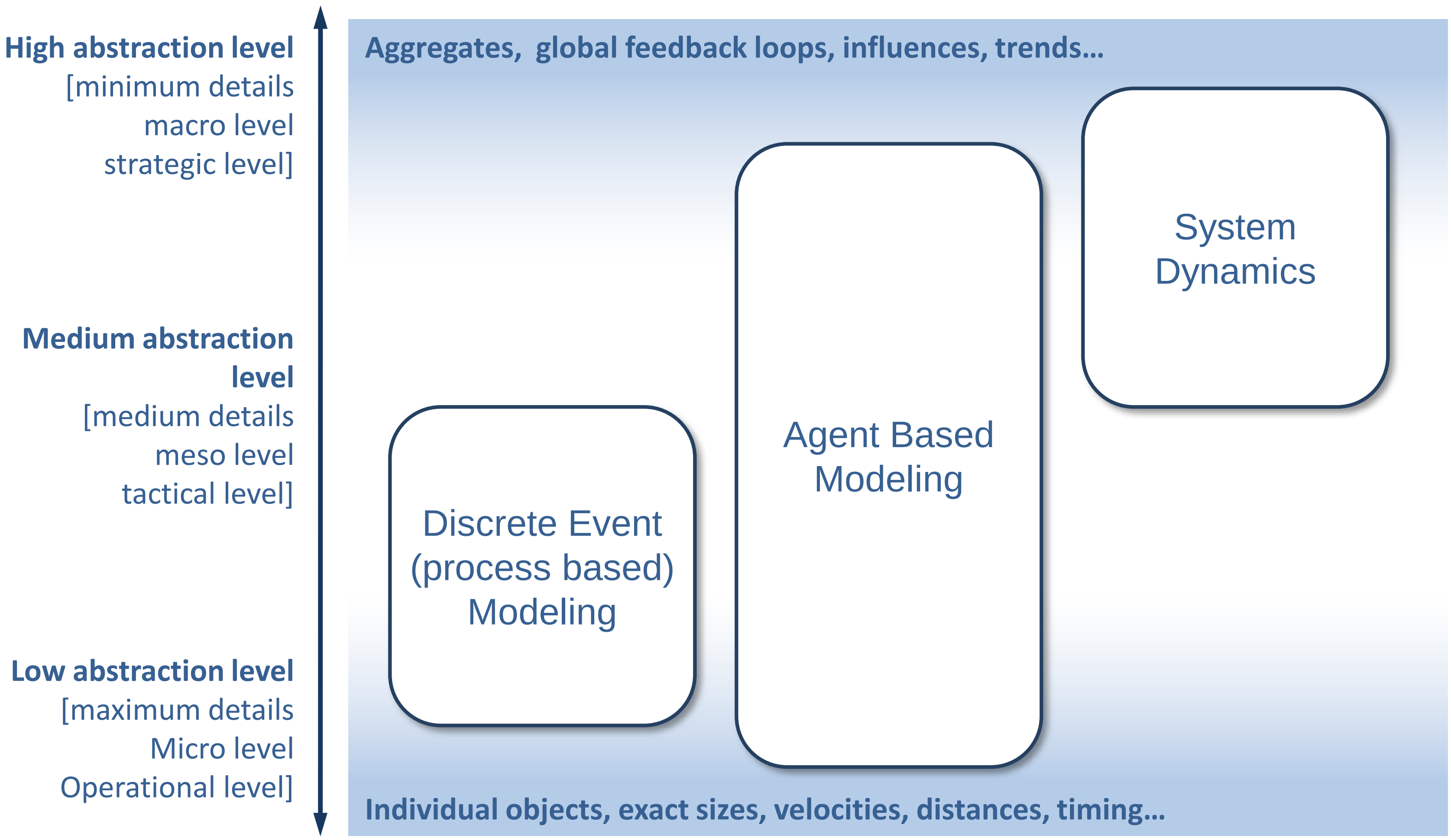
Low abstraction level
[maximum details
Micro level
Operational level]

Aggregates, global feedback loops, influences, trends...

- Market and competition
- Social systems
- Project and product management
- Ecosystems
- HR dynamics
- Health economics
- Energy supply networks
- Asset management
- Supply chains
- Transportation
- Healthcare
- Business processes
- Manufacturing
- Service systems
- Battlefield, command and control
- Warehouse logistics
- Pedestrian dynamics
- Computer hardware
- Physical control systems

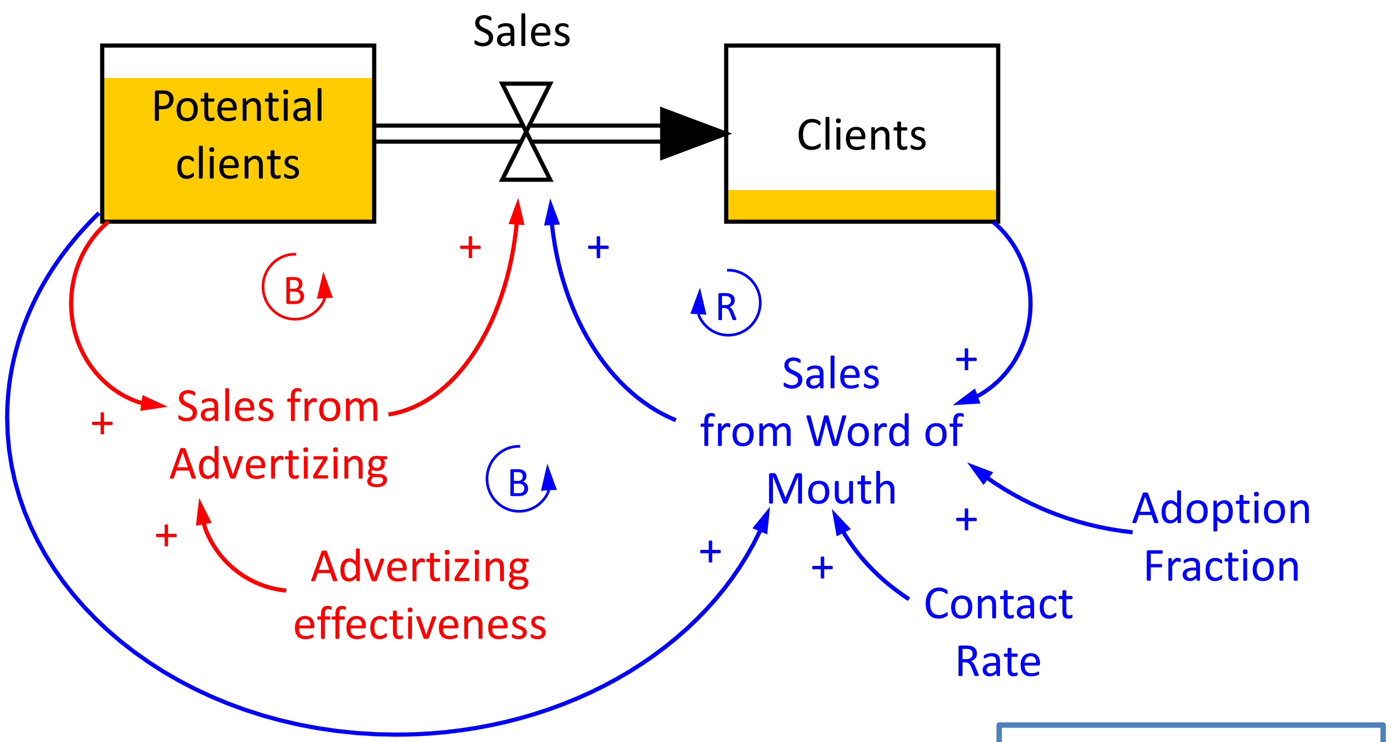
Individual objects, exact sizes, velocities, distances, timing...

Methods in simulation modeling



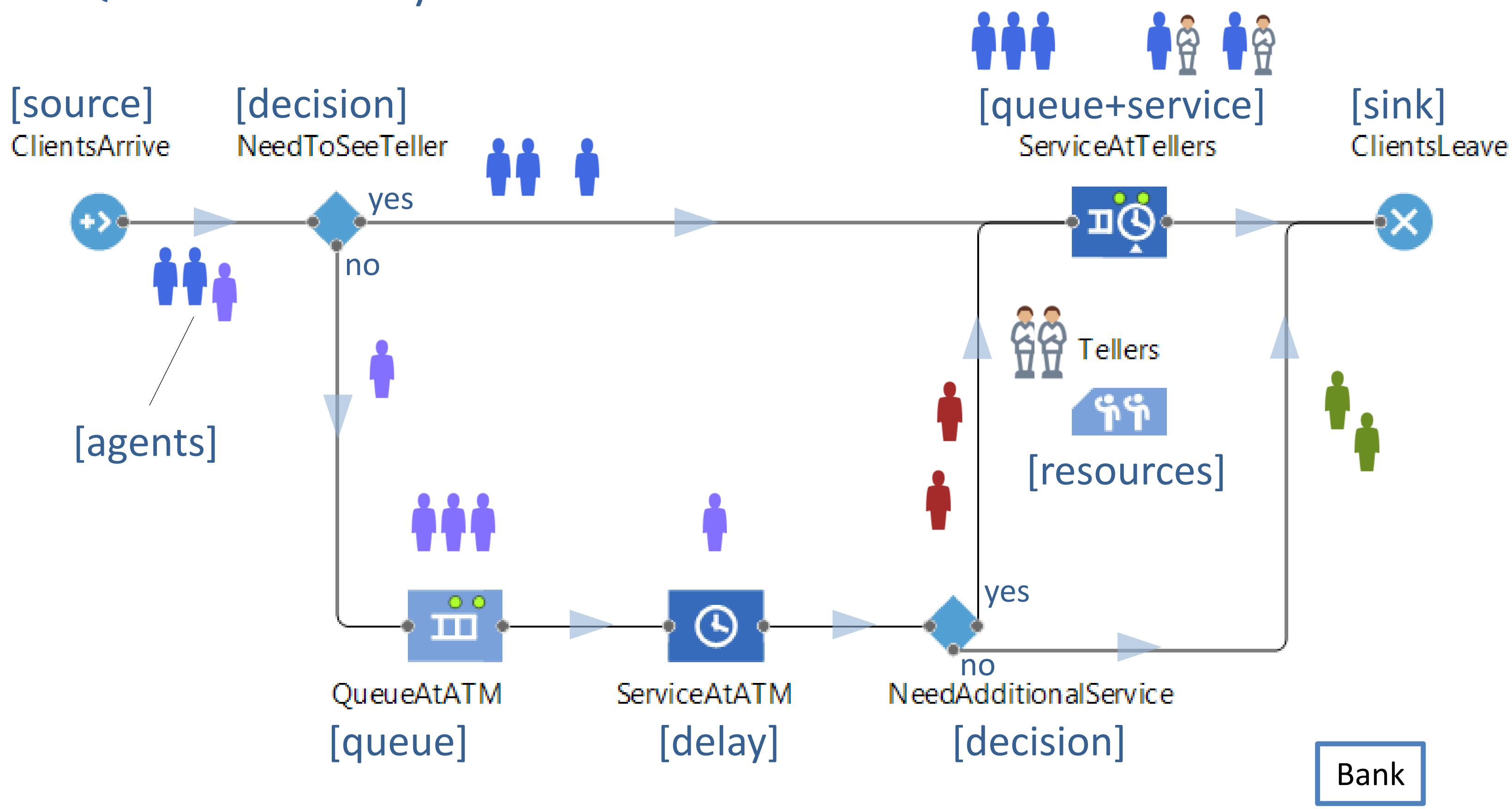
System Dynamics Jay Forrester '50s

- Stocks, flows
  - Interacting feedback loops



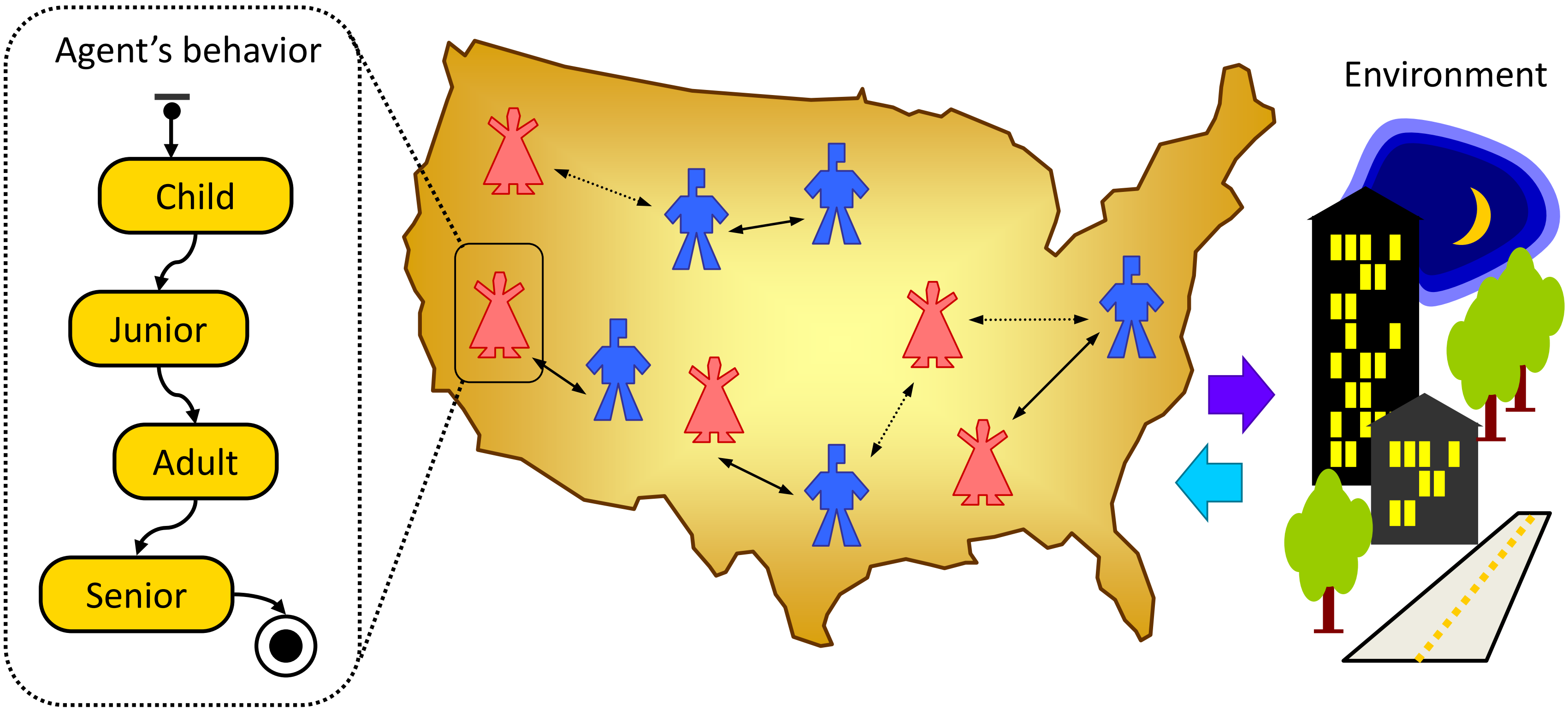
Discrete event modeling. G. Gordon '60s

- Agents and resources. Flowchart diagram
  - Queues and delays



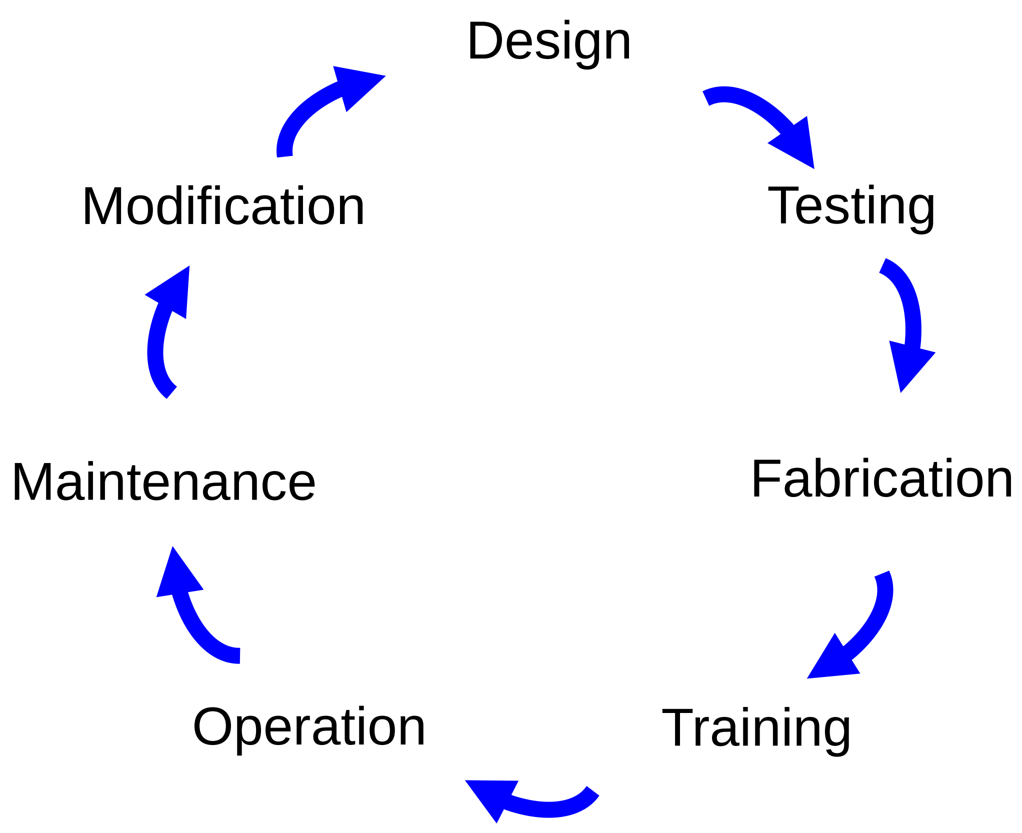
Agent based modeling

- We focus on individual objects and describe their local behavior, local rules
  - Sometimes, we also model the dynamics of the environment



When to simulate - the circle of life

Simulation



Characteristics of a simulation model

- Takes random (stochastic) behavior into account
- May models each agent moving through a system
- Handles complex interactions
- Not all system details are modeled
- Abstracts the system to an appropriate level
- Compresses time
- Can animate system explicitly or conceptually

Model input data

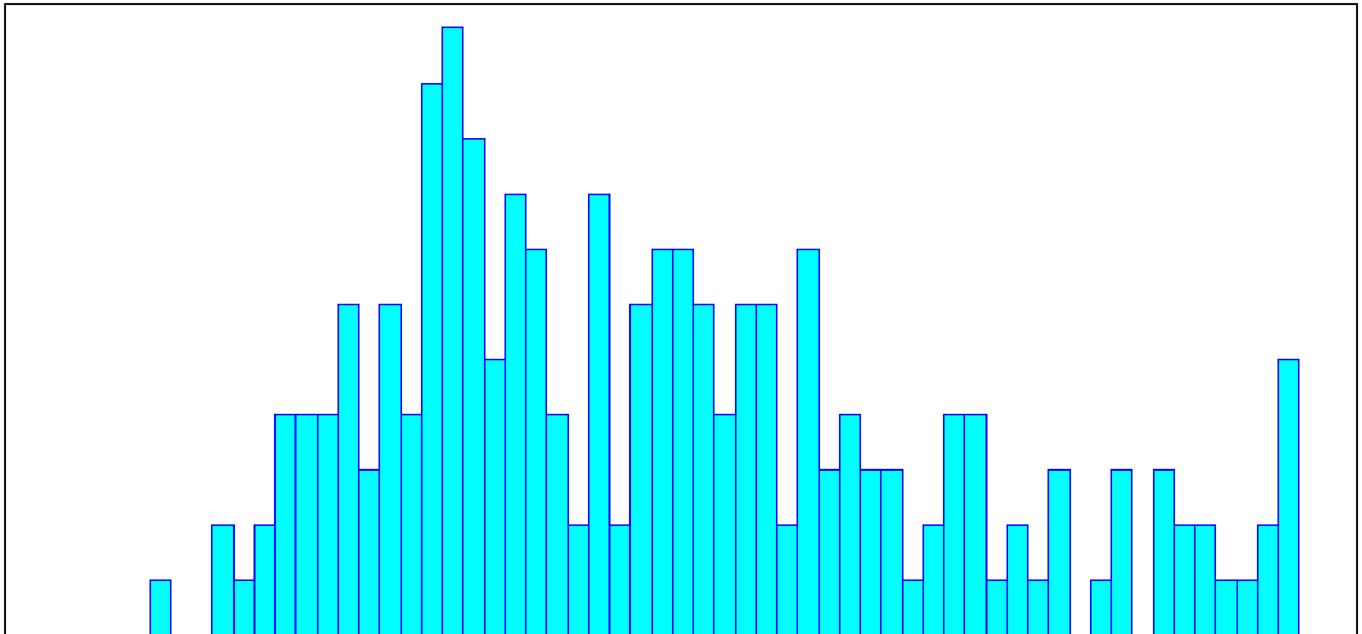
- On-line data
- Direct observation, time study
- Check sheets
- Analysis of similar process
- Review of manual documentation
- Short-term automated data collection methods

- Example: Lab test turn around time

Input data example: CBC turn-around time

Test Code: CBC				
Test Name: Complete Blood Count				
Collection Time	Receive Time	Result Time	Result - Receive	Result - Collection
0000	0031	0045	14	45
0001	0012	0021	9	20
0001	0011	0029	18	28
0001	0010	0030	20	29
0001	0018	0032	14	31
0005	0023	0036	13	31
0008	0022	0048	26	40
0010	0030	0047	17	37
0013	0039	0049	10	36
0035	0039	0055	16	20
0100	0113	0122	9	22
0102	0123	0136	13	34
0120	0136	0156	20	36
0124	0133	0141	8	17

Input data example: CBC data histogram



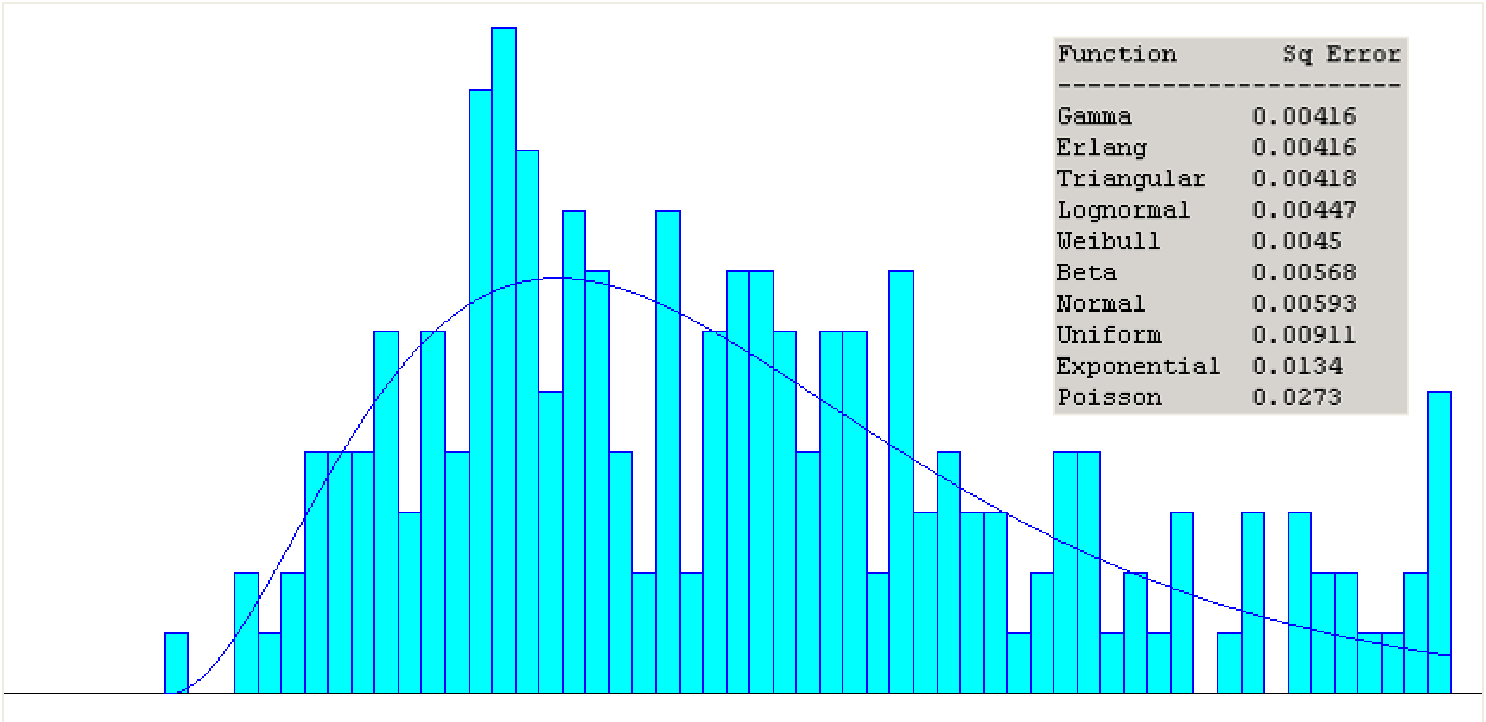
Data Summary

Number of Data Points = 205
Min Data Value = 5
Max Data Value = 59
Sample Mean = 28.9
Sample Std Dev = 12.9

Histogram Summary

Histogram Range = 4.5 to 59.5
Number of Intervals = 55

Input data example: CBC data – best fit function



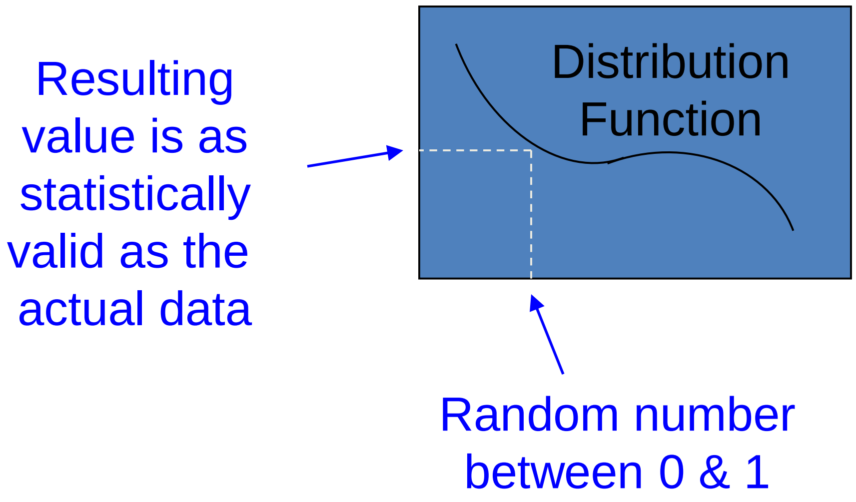

```
        Distribution Summary

Distribution:   Gamma
Expression:     4.5 + GAMM(7.66, 3.19)
Square Error:   0.004161

Chi Square Test
  Number of intervals   = 16
  Degrees of freedom    = 13
  Test Statistic        = 16.9
  Corresponding p-value = 0.216
```

Input data example: CBC data – use in the model

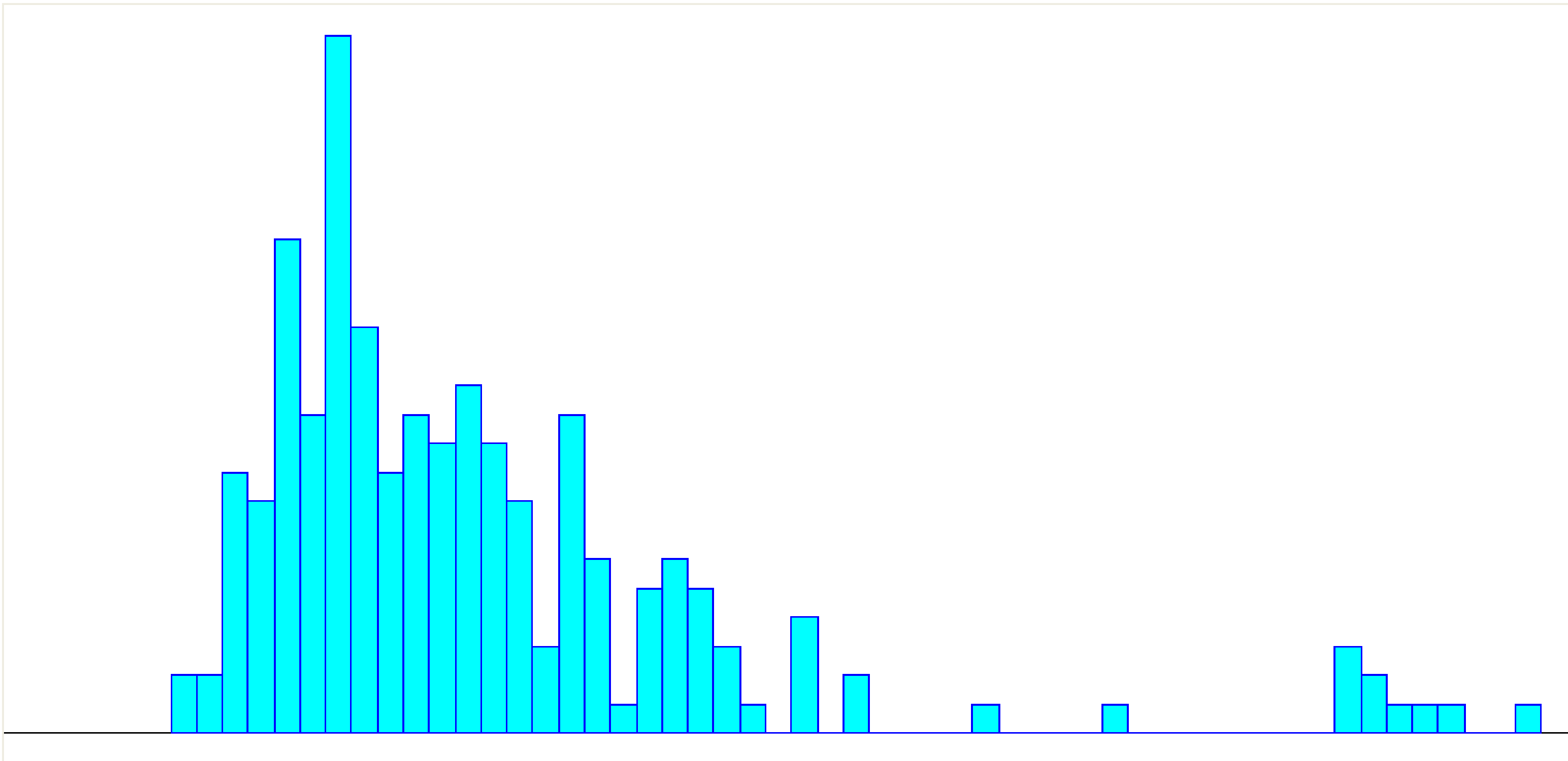
Simulation uses this probability density function {(4.5 + GAMM(7.66, 3.19)} each time a CBC is performed

Random number (x) drives the generation of a CBC time

Simulated time is statistically equal to actual time



Input data example: CBC data – bimodal?

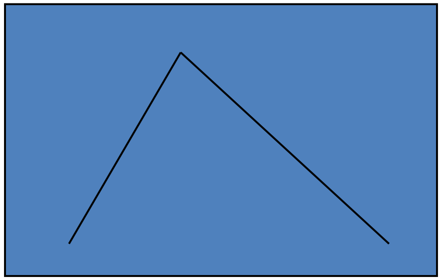


Data Summary

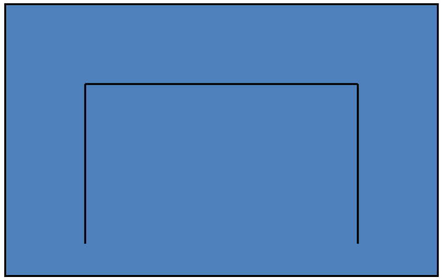
Number of Data Points = 205
Min Data Value = 2
Max Data Value = 54
Sample Mean = 13.6

Distributions: Types

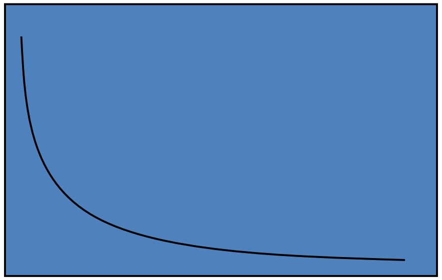
Triangular



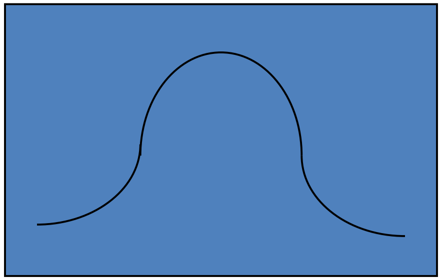
Uniform



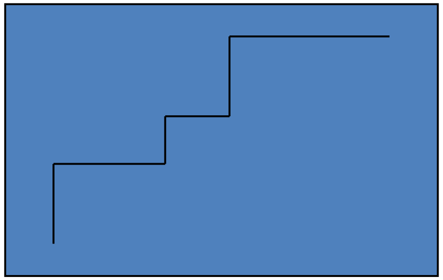
Exponential



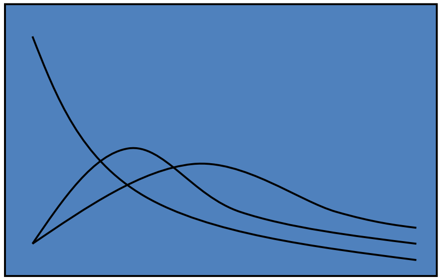
Normal



Discrete



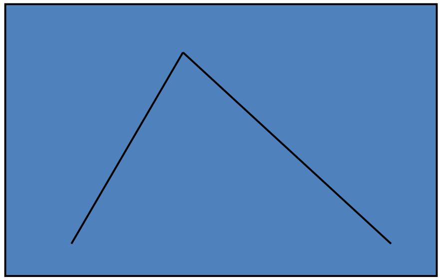
Erlang



Others: Beta, Gamma, Johnson, Weibull, Poisson, Continuous, Lognormal, User Defined

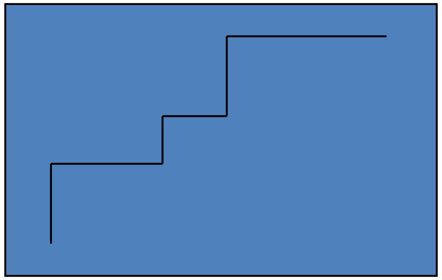
Distributions: Typical uses

Triangular



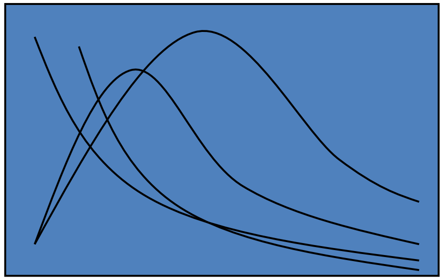
Distribution not known, but can estimate or guess minimum, maximum, and most likely

Discrete



Assignments such as attributes, sequences, batch sizes

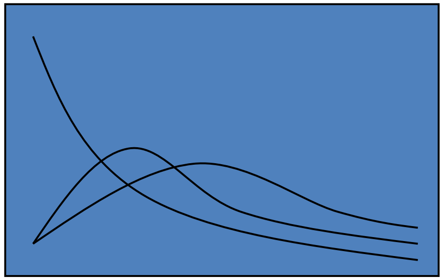
Weibull



Time between failures

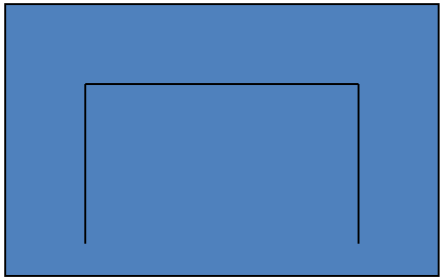
Distributions: Typical uses

Erlang & Gamma



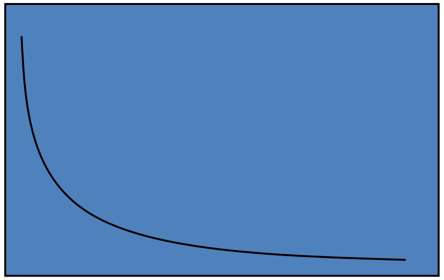
Often used to represent the time it takes to complete a task

Uniform



All values equally likely.
No information other than range available

Exponential

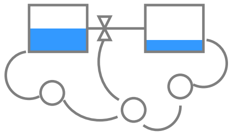


Inter-event times in arrivals and breakdowns

Simulation modeling software

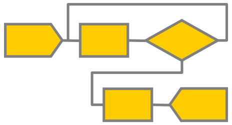
- A partial list of tools and the modeling approaches they support

System dynamics



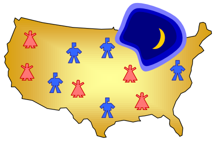
AnyLogic
iThink
PowerSim
Stella
VenSim

Discrete event modeling



AnyLogic
Arena
AutoMod
Enterprise Dynamics
ExtendSim
FlexSim
PROMODEL
Simio
SimProcess

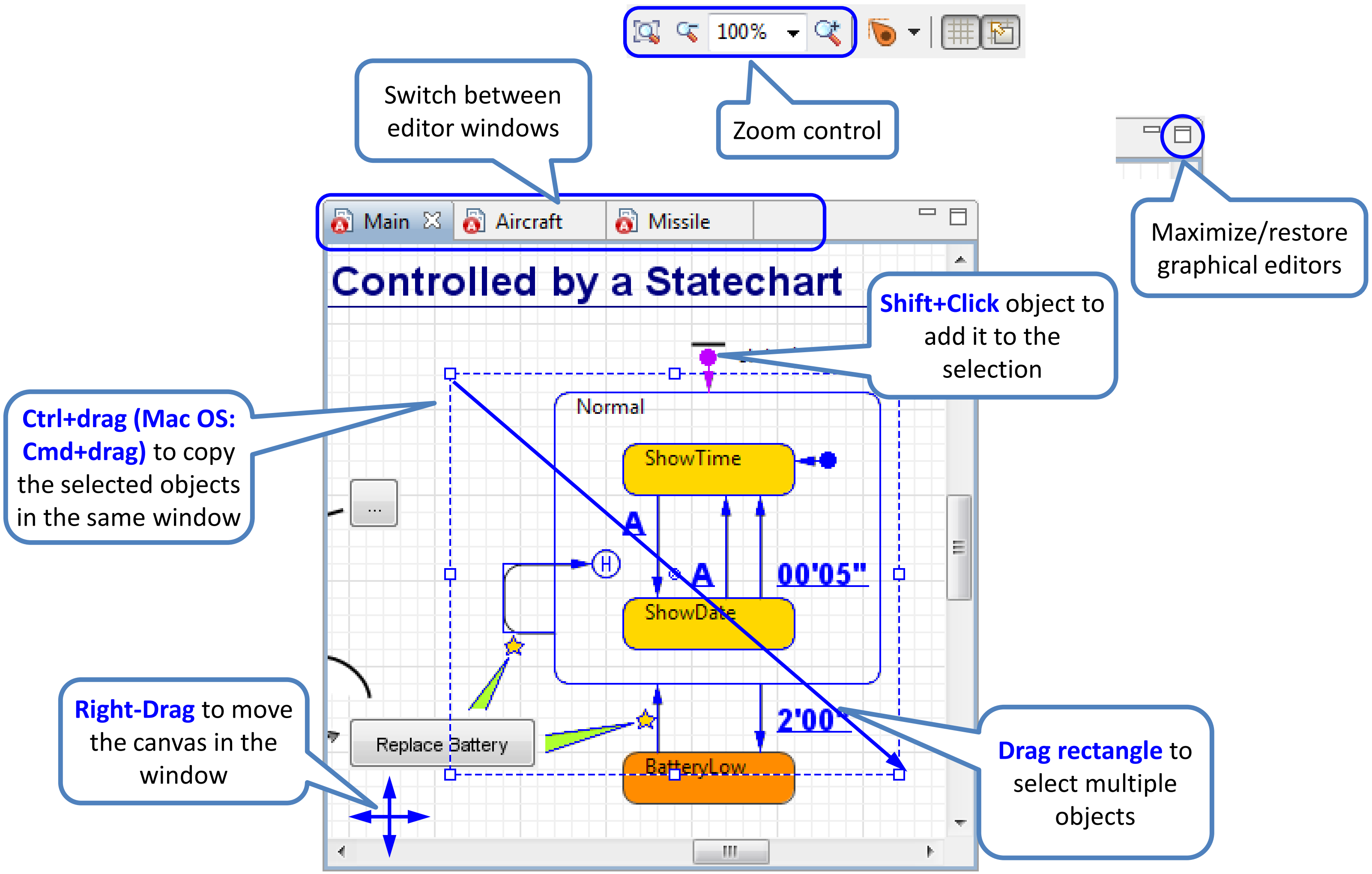
Agent based modeling



AnyLogic
Academic tools:
- ASCAPE
- NetLogo
- RePast
- Swarm

The Model Development Environment

Graphical Editor – Selecting & Copying



Properties view

- The **Properties** view allows you to view and modify the selected item's properties (you select an item by clicking it in the graphical editor, or in **Projects** view).

Legend:

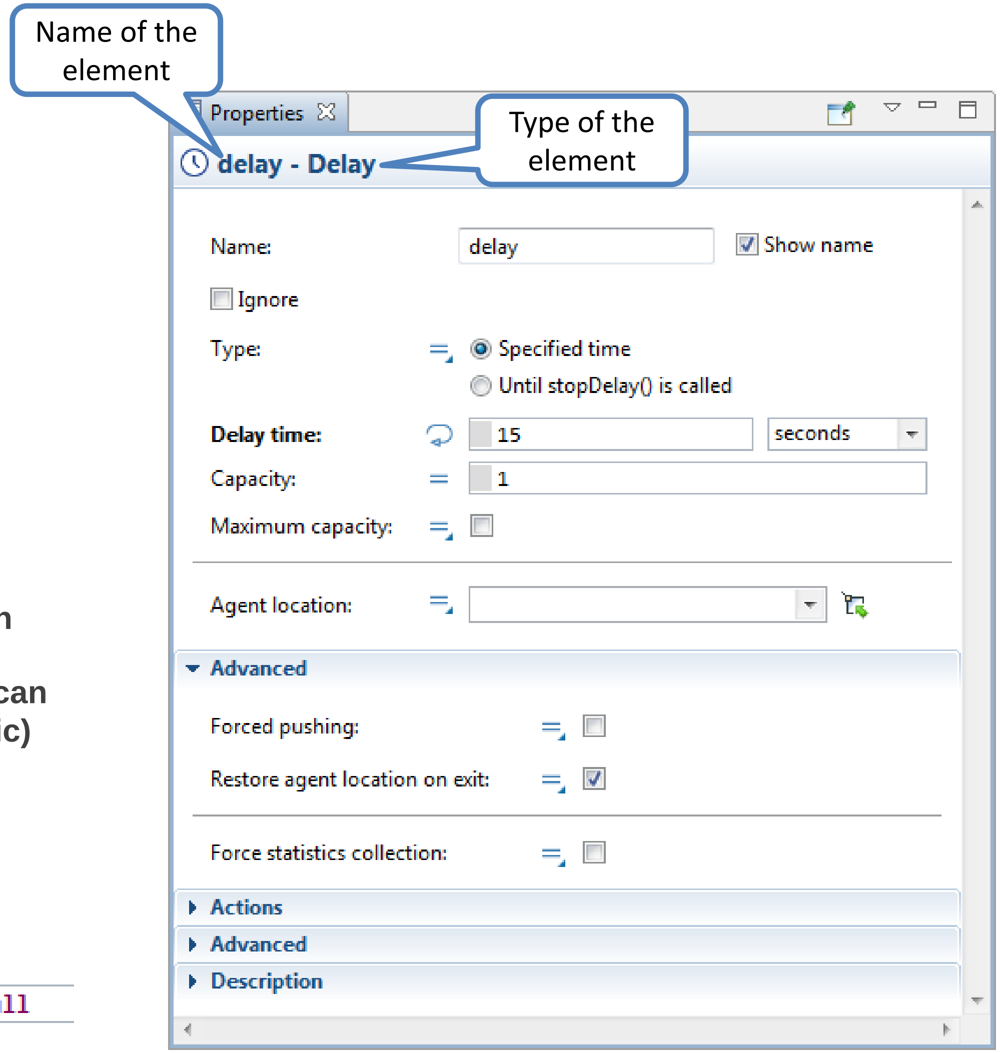
Static value

Dynamically evaluated expression

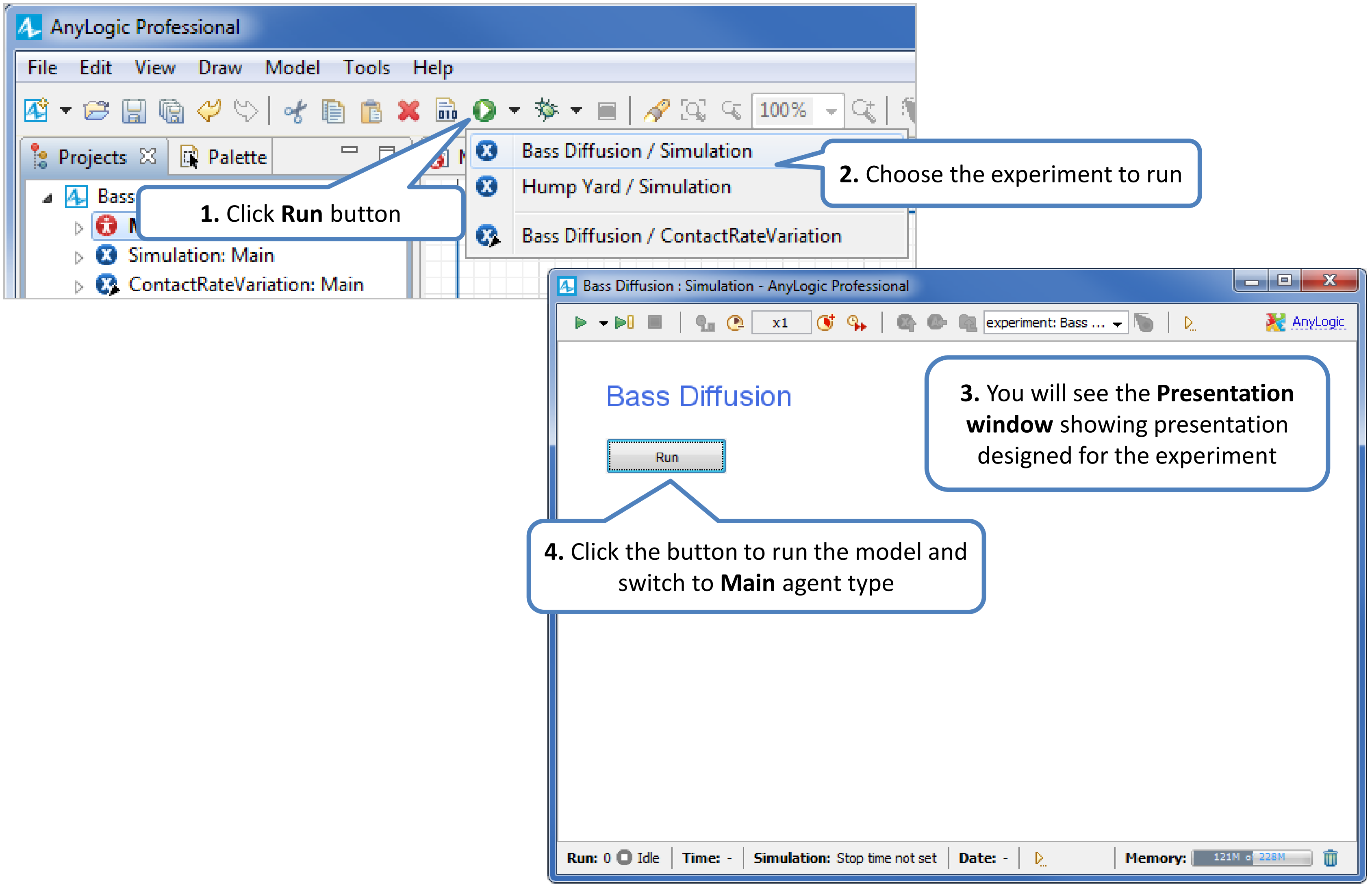
Small triangle indicates that you can switch between design-time (static) and run-time (dynamic) values

Fill color: red

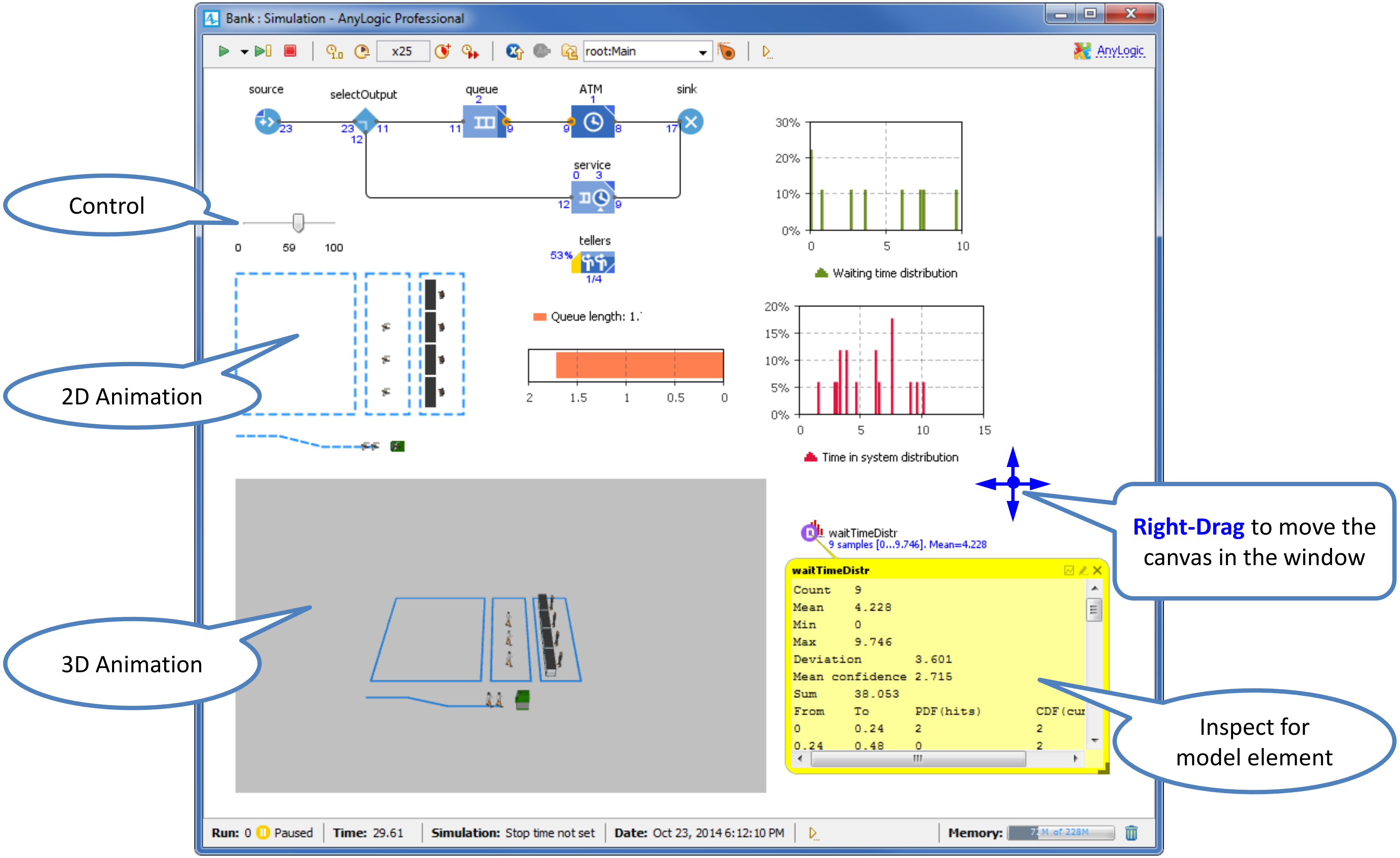
Fill color: `line.isVisible() ? red : null`



Running the Model

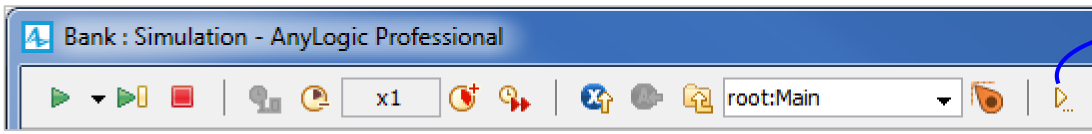


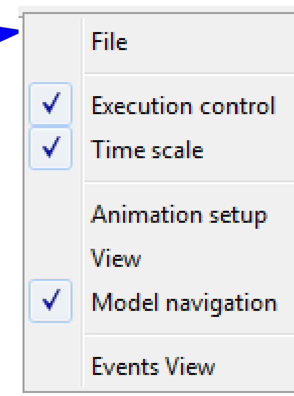
Presentation Window



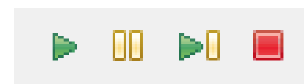
Major Toolbar Commands

- Toolbars and status bar can be customized





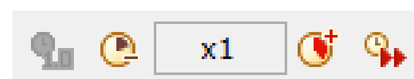
- Execution control



Only one is displayed at a time, depending on the model state {
Run from the current state:
Pause:

Make a step:
Terminate execution:

- Time scale



Set real time mode at default scale:

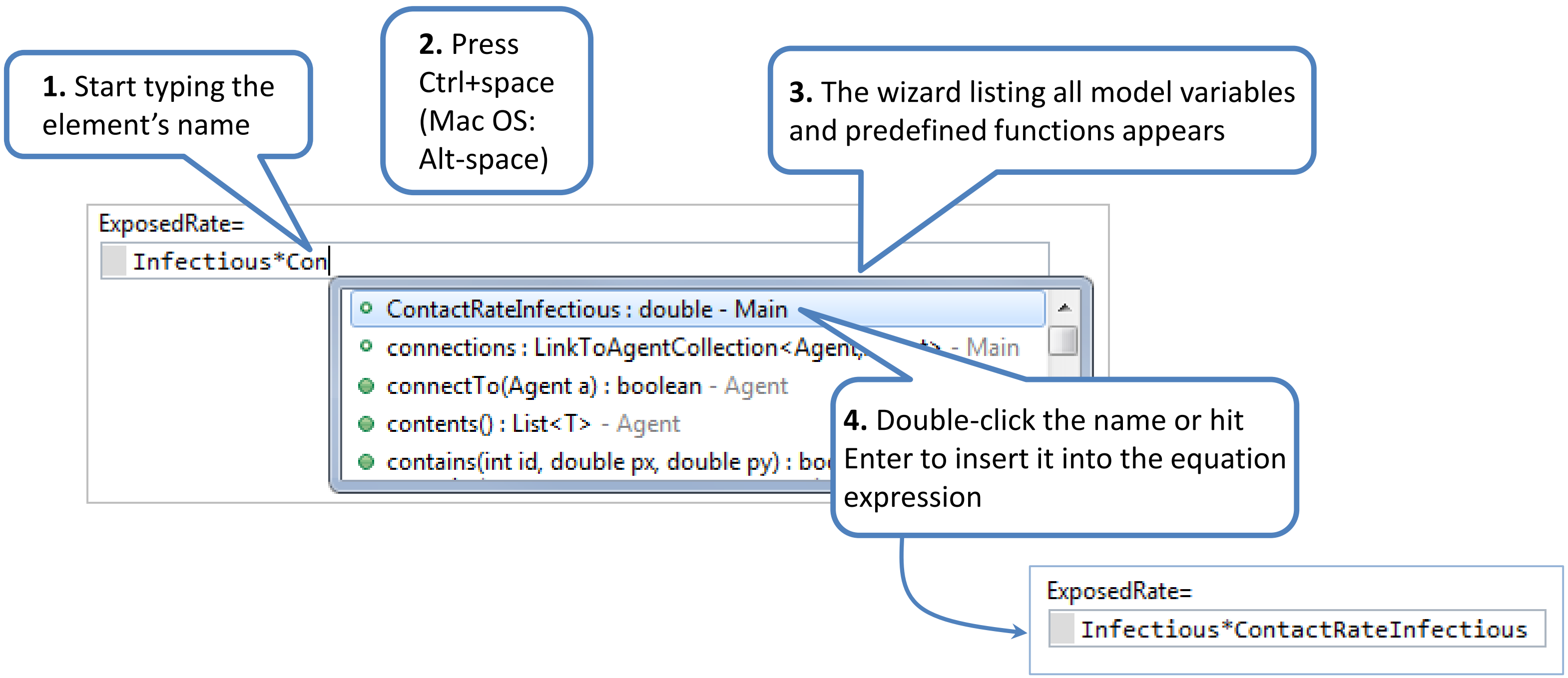
Real Time mode only {
Decrease execution speed:
Select execution speed factor:
Increase execution speed:

Toggle Real/Virtual time modes:

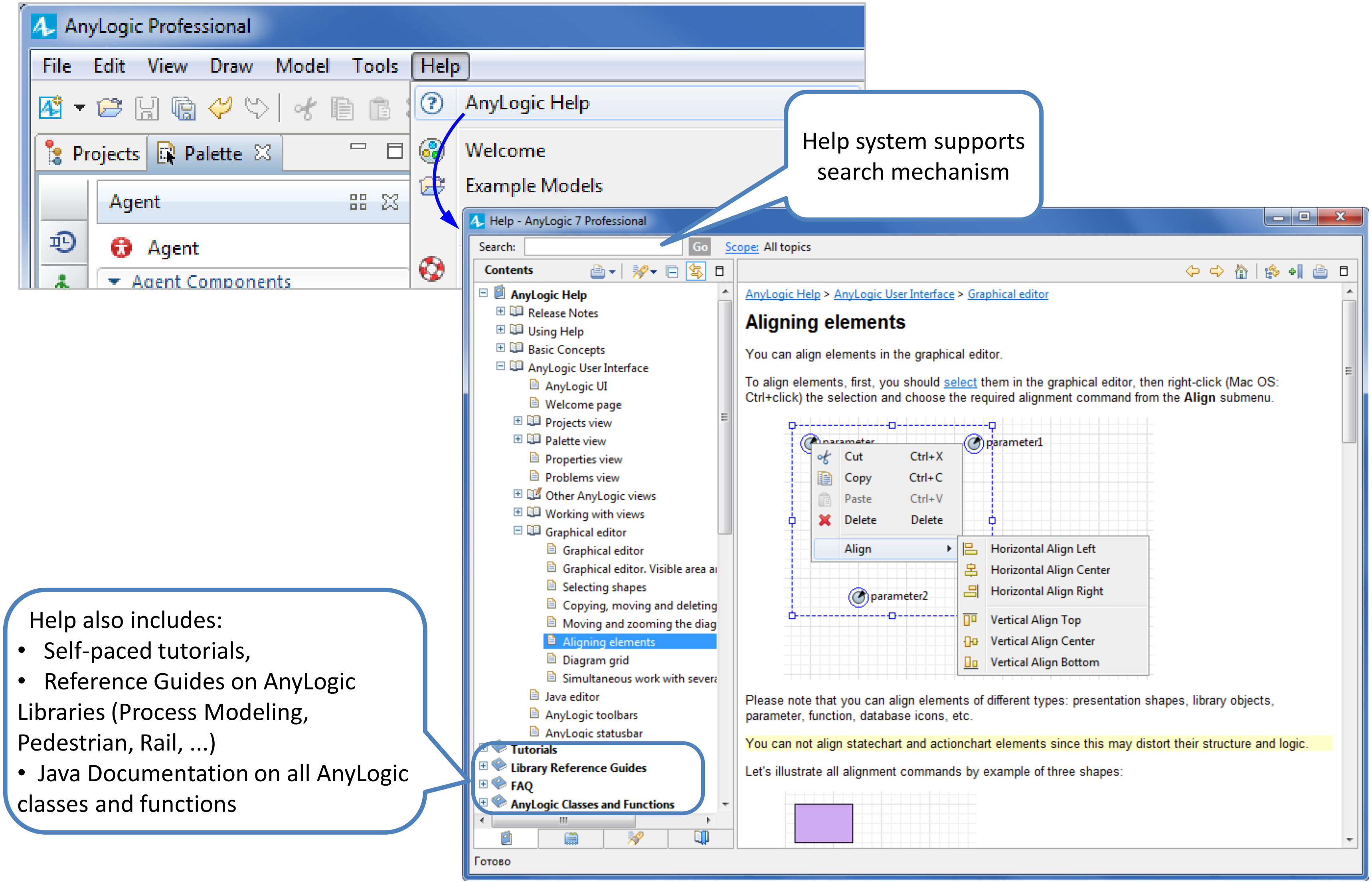
Code Completion Master

Intelli-sense mechanism:

The wizard looks as a list, containing variables, parameters, and functions. You can simply select the name in the list, and it will be inserted in the expression automatically.



Help



Help system supports search mechanism

Help also includes:

- Self-paced tutorials,
- Reference Guides on AnyLogic Libraries (Process Modeling, Pedestrian, Rail, ...)
- Java Documentation on all AnyLogic classes and functions

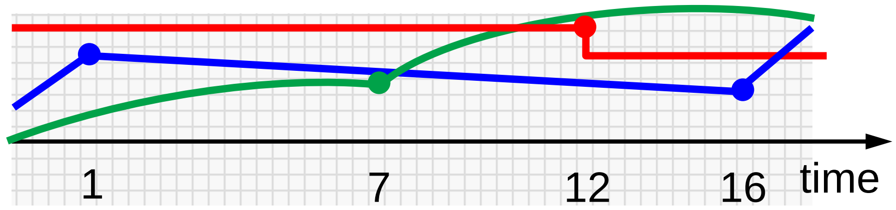
Discrete Event Modeling

Event

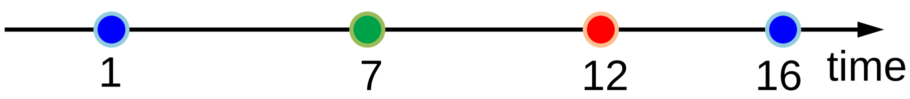
- We consider only “important moments” in the system’s lifetime, which are called *events*.
  - Any change in the model may happen only as a result of an event
- Examples:
  - Customer arrives at the bank office
  - Bill finishes processing
  - Amount of raw material reaches the minimum level
- Event:
  - Takes zero model time
  - Causes changes in the model
  - May schedule or cancel other events in the future

Time as event order – Discrete Time

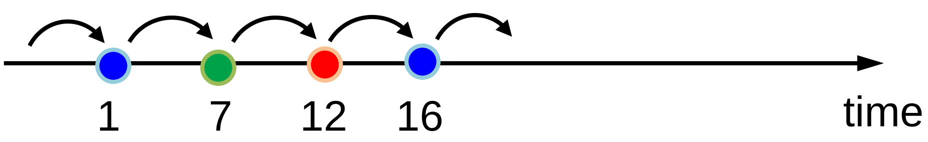
- We consider only a sequence of instant “discrete” events, while nothing happens in between



- No “continuous-time” processes



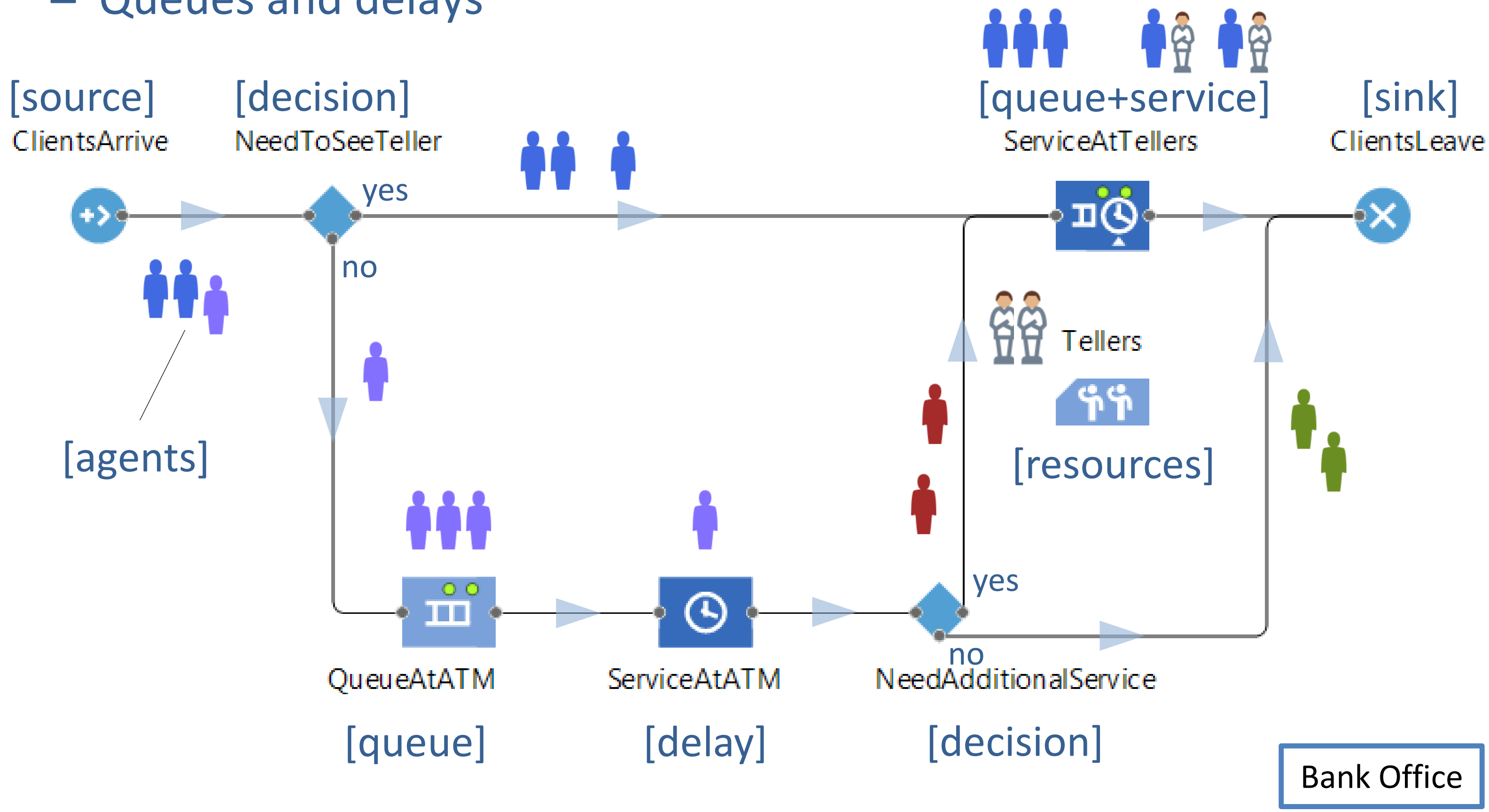
- Model time “jumps” from one event to another



This is [Discrete Event Modeling]

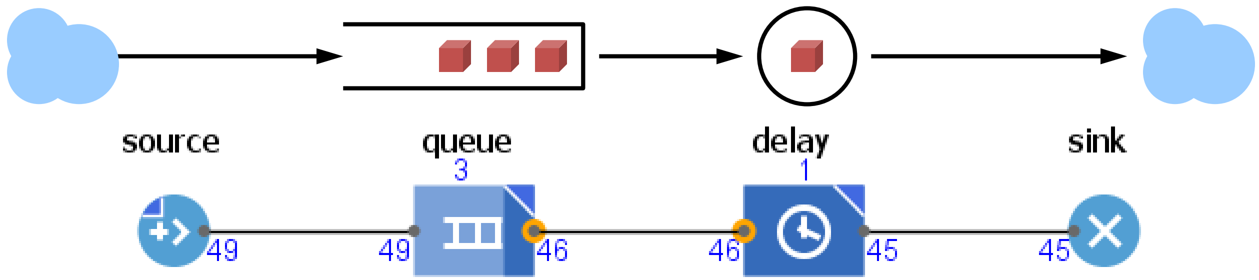
Discrete event modeling. G. Gordon ’60s

- Agents and resources. Flowchart diagram
  - Queues and delays



The simple process based model

- Define the process with Process Modeling Library blocks with a simple drag and drop



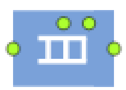
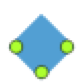
Arrivals defined by: Rate
Arrival rate: 1
Limited number of arrivals:
New agent: Agent
Location of arrival:
On exit:

Capacity: 100
On enter:
On at exit:
On exit:
Enable exit on timeout:
Agent location:

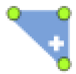
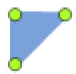
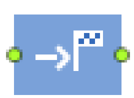
Delay time: triangular(0.5, 1, 1.5)
Capacity: 1
On enter:
On exit:
Agent location:

On enter:

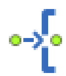
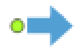
Process Modeling Library. Essential blocks (1/3)

Block name	Icon in graphical editor	Description
Source		Generates agents. Is usually a starting point of a process model.
Sink		Disposes incoming agents. Is usually an end point in a process model.
Queue		Stores agents in the specified order.
Delay		Delays agents by the specified delay time.
SelectOutput		Forwards the agent to one of the output ports depending on the condition.
SelectOutput5		Routes the incoming agents to one of the five output ports depending on (probabilistic or deterministic) conditions.

Process Modeling Library. Essential blocks (2/3)

Block name	Icon in graphical editor	Description
Conveyor		Simulates conveyor. Moves agents at a certain speed, preserving order and space between them.
Split		Creates a new agent (copy) of the incoming agent.
Combine		Waits for two agents, then produces a new agent from them.
Batch		Accumulates agents, then outputs them contained in a new agent.
Unbatch		Extracts all agents contained in the incoming agent and outputs them.
MoveTo		Moves an agent from its current location to new location.

Process Modeling Library. Essential blocks (3/3)

Block name	Icon in graphical editor	Description
RestrictedAreaStart		Using these blocks you can limit the number of agents in a part of flowchart between corresponding RestrictedAreaStart and RestrictedAreaEnd blocks.
RestrictedAreaEnd		
TimeMeasureStart		TimeMeasureEnd and TimeMeasureStart compose a pair of blocks measuring the time the agents spend between them.
TimeMeasureEnd		
Exit		Takes the incoming agents out of the process flow and lets the user to specify what to do with them.
Enter		Inserts the (already existing) agents into a particular point of the process model.

Parameters of Process Modeling Library blocks

- Simple static parameters:
  - Evaluated once, but may be changed during the model execution
- Dynamically evaluated expressions (dynamic parameters):
  - Evaluated each time they are needed, e.g. each time the delay time, the speed or other property of an agent needs to be obtained
  - The corresponding agent is accessible as "agent", etc.
- Dynamically executed code pieces (code parameters):
  - Evaluated each time a certain event occurs at the object: the agent enters/exits it, conveyor stops, etc.

Capacity = 100

Delay time: exponential(1)

Condition: agent.type == VIP

Speed: agent.cruiseSpeed

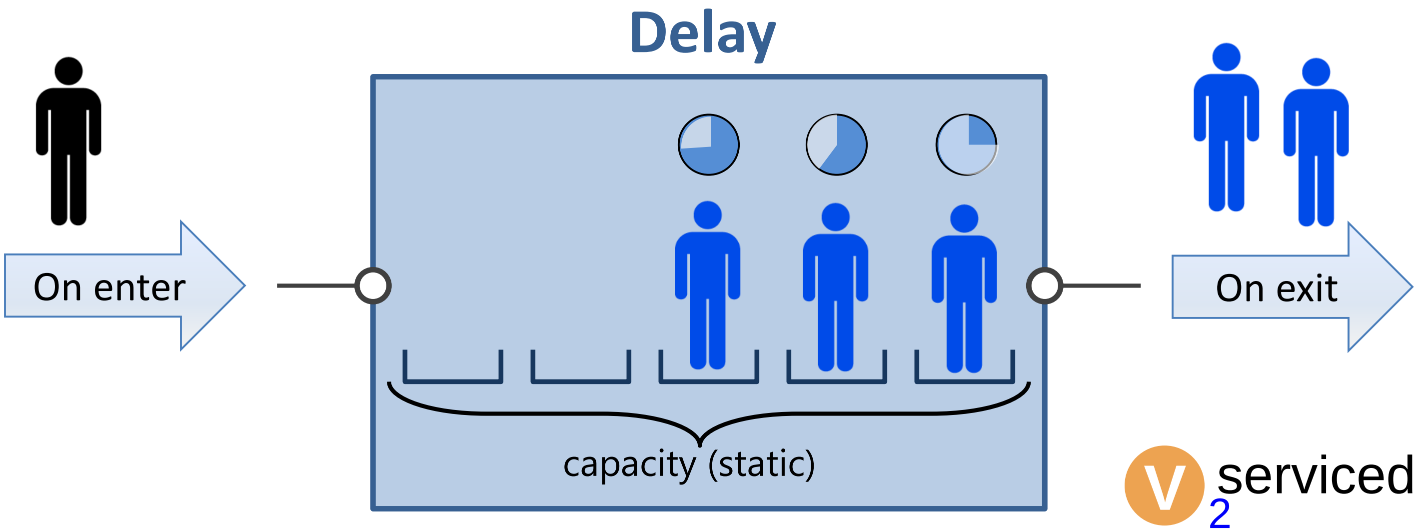
Actions

On exit: agent.setColor(red);

On enter:

```
if( agent.airline == "AF")
  agent.destination = 
gate17;
```

Parameters. Examples



Capacity: 5

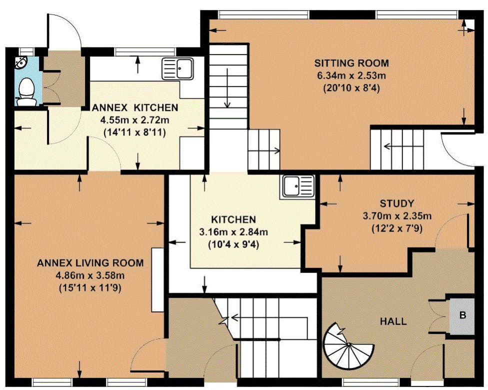
Delay time: uniform(2, 10)

On enter: agent.setColor(blue);

On exit: serviced++;

- To change a value of the static `capacity` parameter, call: `delay.set_capacity(20);`
- You add a semicolon at the end of each line of Java code in code parameter
- You do not add a semicolon to the end of static/dynamic parameter expression

Resources

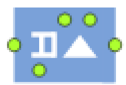
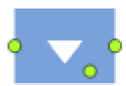
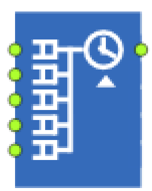

SITTING ROOM
6.34m x 2.53m
(20'10 x 8'4)
ANNEX KITCHEN
4.55m x 2.72m
(14'11 x 8'11)
KITCHEN
3.16m x 2.84m
(10'4 x 9'4)
STUDY
3.70m x 2.35m
(12'2 x 7'9)
ANNEX LIVING ROOM
4.86m x 3.58m
(15'11 x 11'9)
HALL
B


T-Max

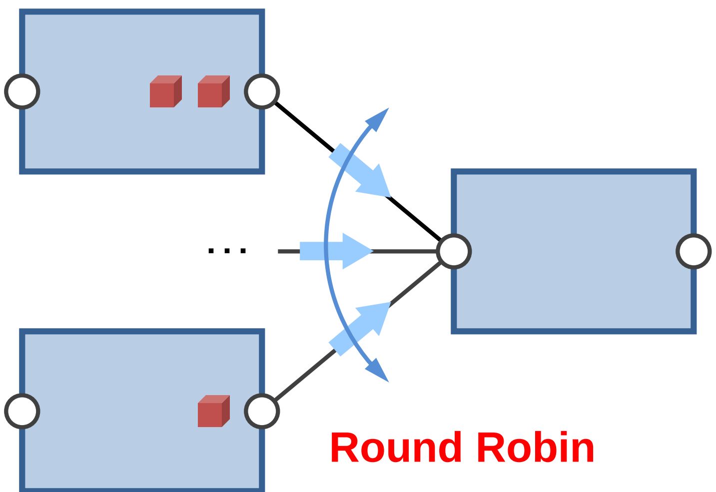
Resource types

- Static (can’t move and can’t be moved): a room, a non-portable equipment, a passage, etc.
- Portable (can’t move on their own, but can be moved): a wheelchair, portable xRay, etc.
- Moving (can move, carry portable resources): a doctor, a nurse, a forklift truck, etc.

PML blocks associated with resources

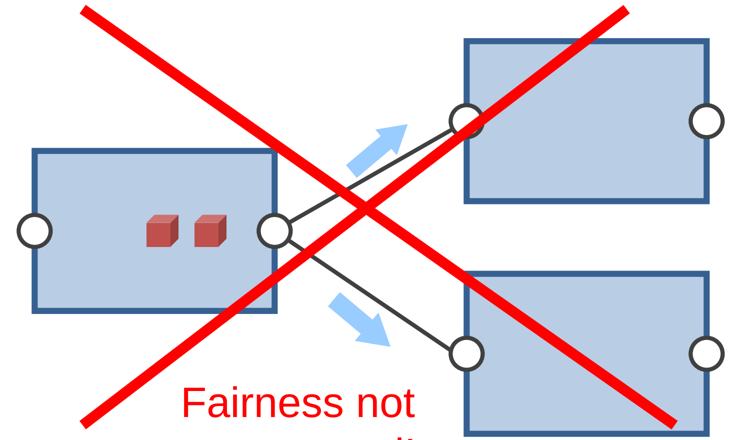
Block name	Icon in graphical editor	Description
ResourcePool		Defines a set of resources of the specified type: how many resources of this type exist in the system, what are they attributes.
Seize		Seizes the number of units of the specified resource required by the agent.
Release		Releases resource units previously seized by the agent.
Service		Seizes resource units for the agent, delays it, and releases the seized units. The block itself is a Seize – Delay – Release sequence of blocks.
Assembler		Assembles a certain number of agents from several sources (5 or less) into a single agent.
ResourceSendTo		Sends a set of portable and/or moving resources to a specified location.

Agent flow at N:1 and 1:N connections

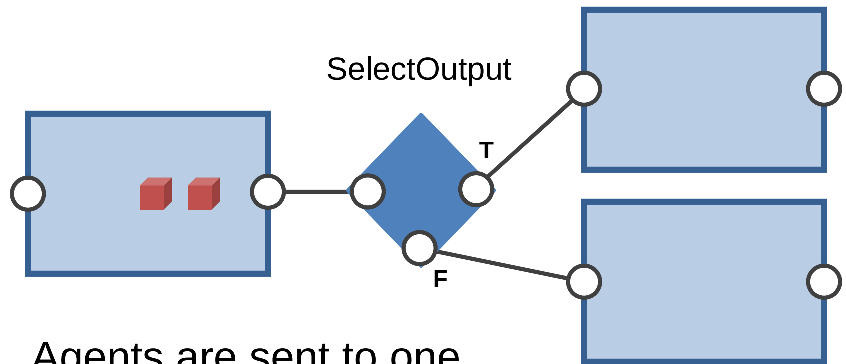


Round Robin

The “competing” outputs are served in round robin manner to ensure fairness



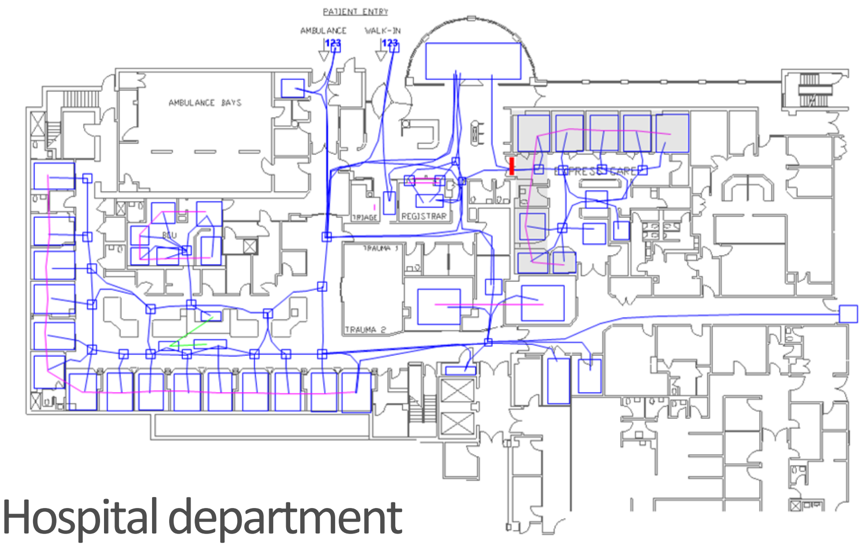
Fairness not guaranteed!



Agents are sent to one of the two output ports depending on (probabilistic or deterministic) condition

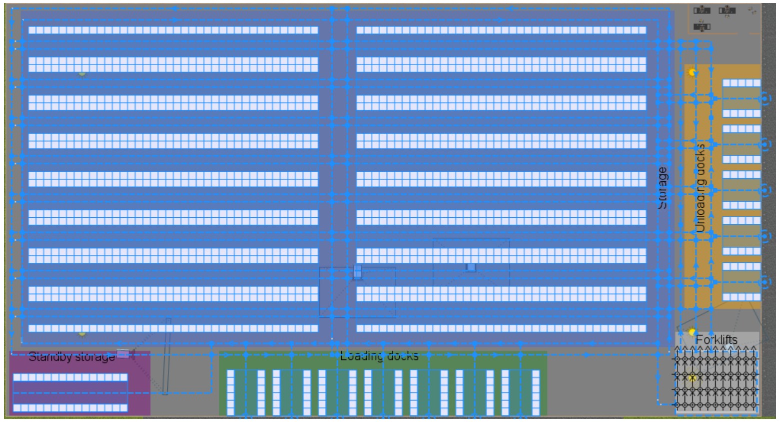
Network-based modeling

- The Process Modeling Library provides better support for certain types of problems where layout is important
  - There is a network of locations and paths between them
  - Agents and resources move along the paths, route lengths matter



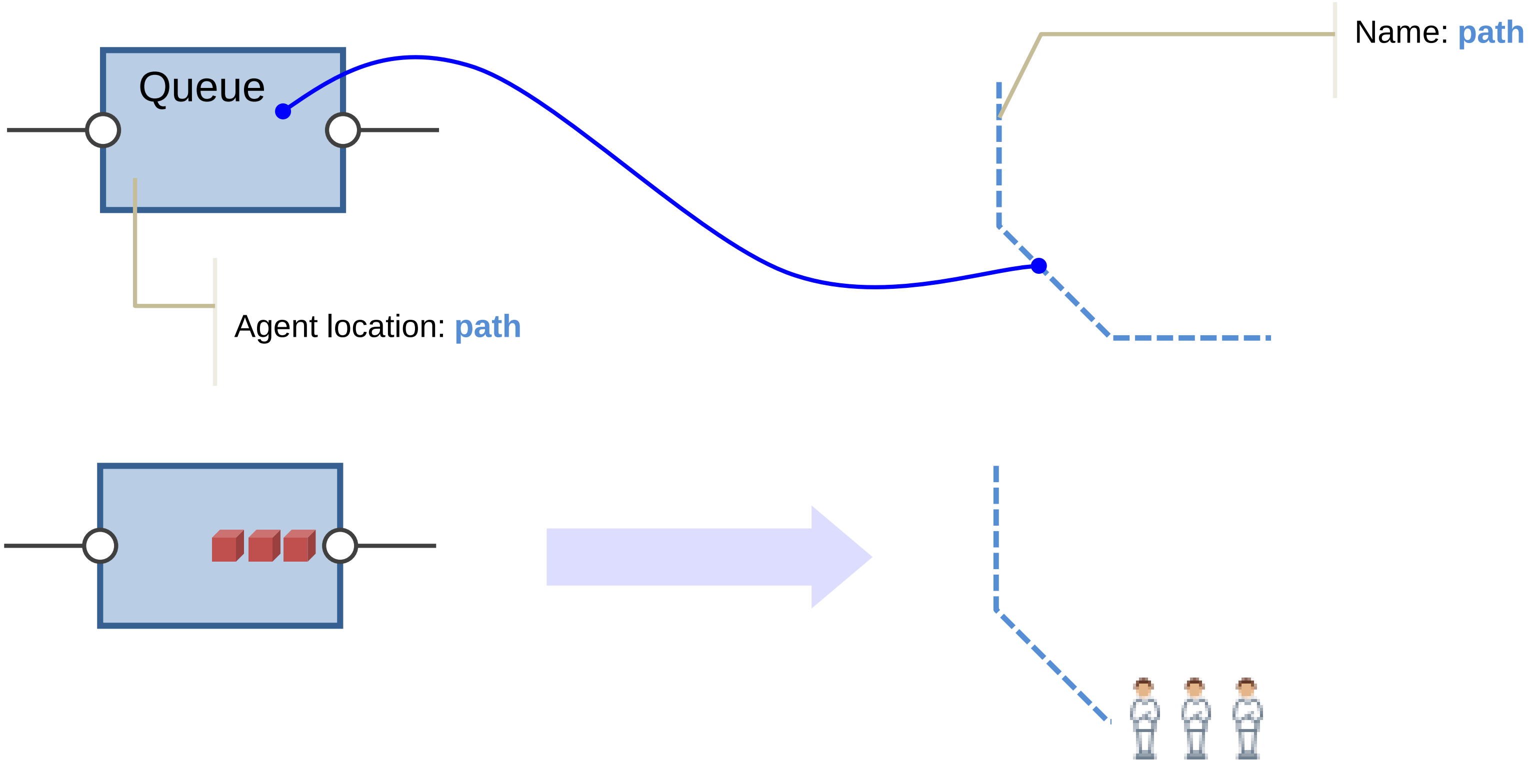
Hospital department

Warehouse



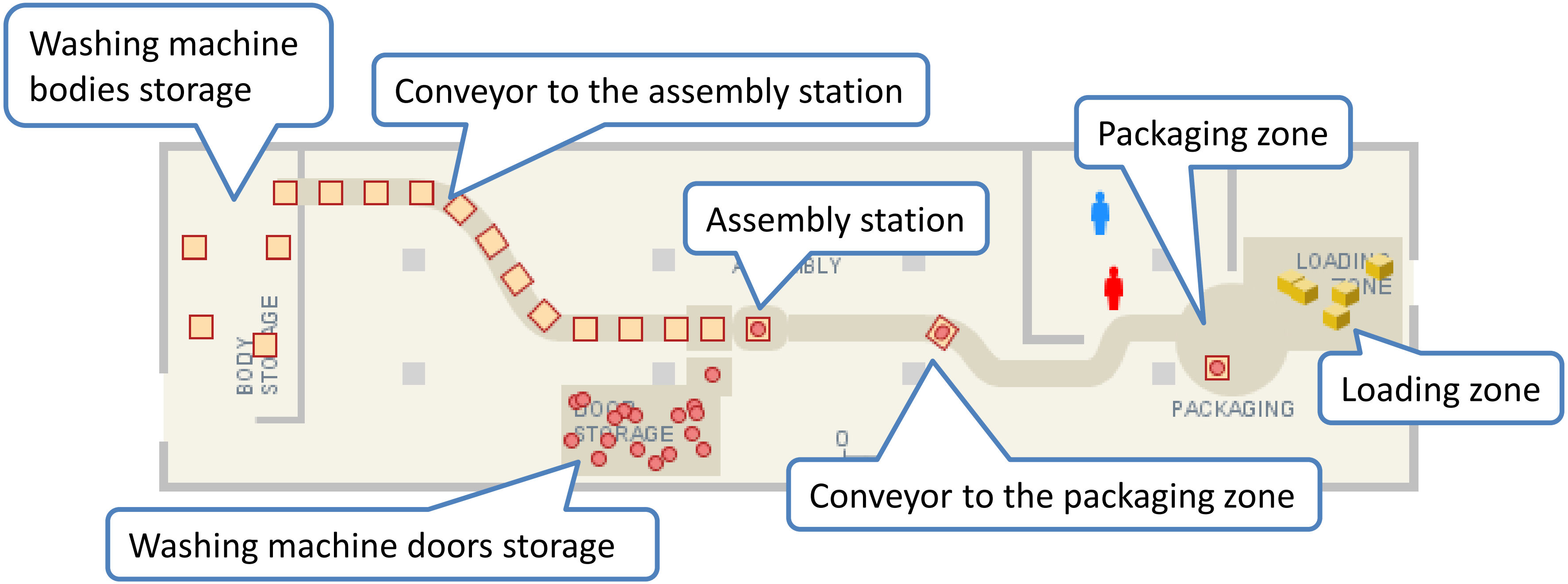
Animation of Process Modeling Library models

You can associate any object that contains agents with a **space markup** shape (**path** or **node**) defining the location of agent animations.



The Factory Model

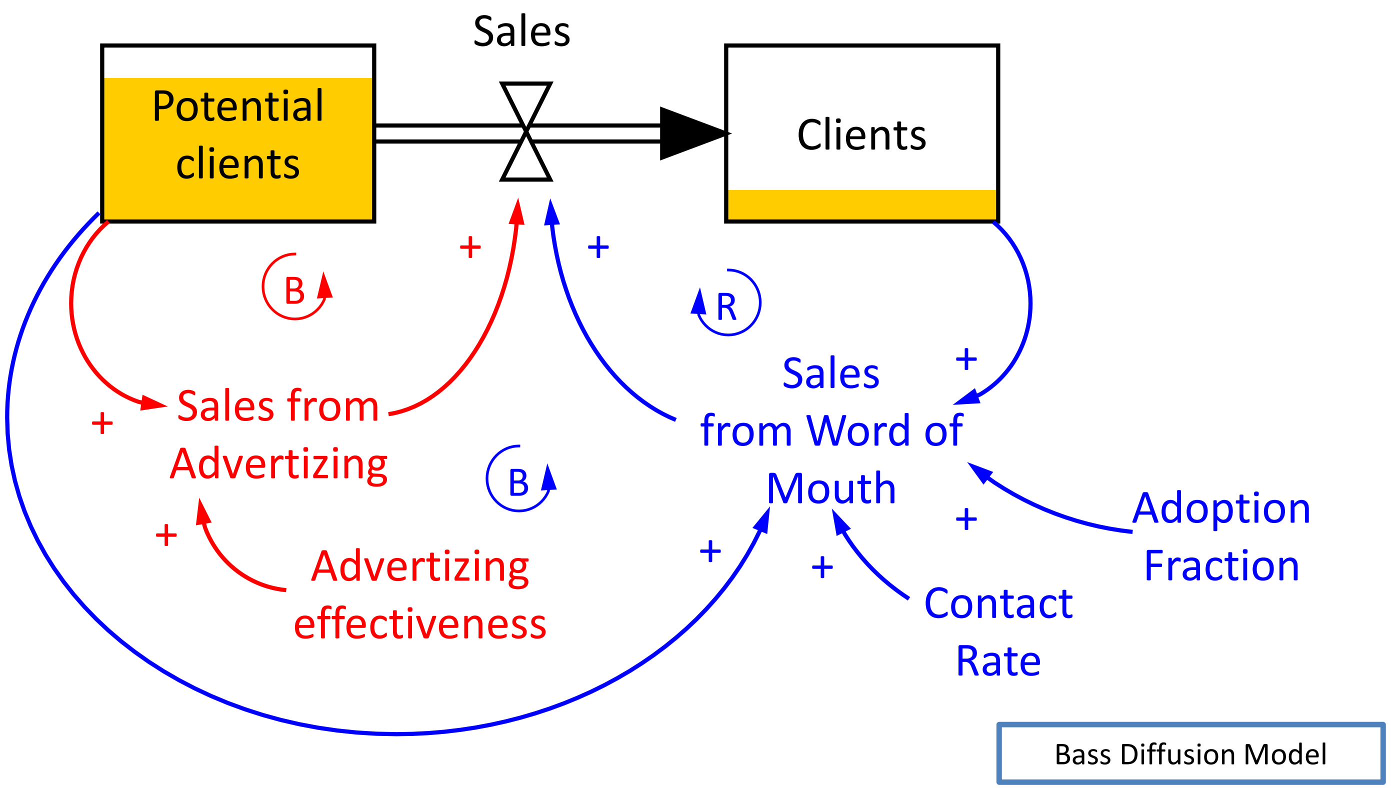
The Factory Model



System Dynamics Modeling

System Dynamics Jay Forrester '50s

- Stocks, flows
  - Interacting feedback loops



System Dynamics Jay Forrester '50s

- Stocks, flows
  - Interacting feedback loops

<u>The equivalent mathematical model:</u>

d(Potential clients)/dt = - Sales

d(Clients)/dt = Sales

Sales = Sales from Advertizing + Sales from Word of Mouth

Sales from Advertizing = Potential clients * Advertizing effectiveness

Sales from Word of Mouth =
Clients * Contact Rate *
(Potential clients / (Potential clients + Clients)) * Adoption Fraction

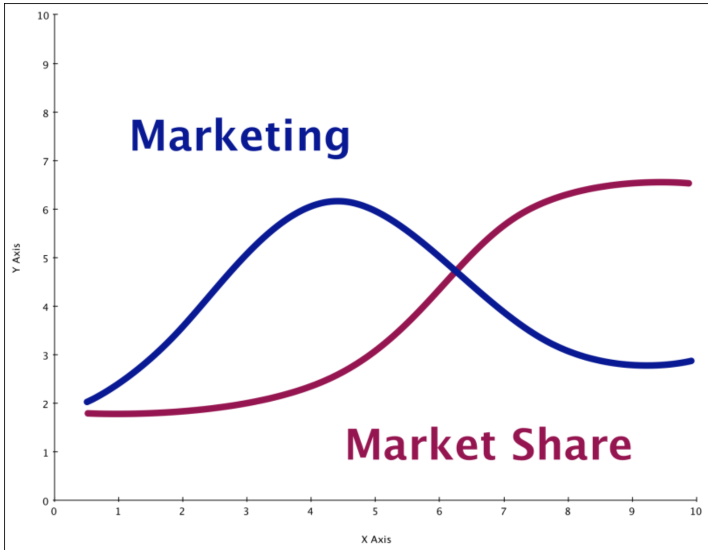
Bass Diffusion Model

System Dynamics excels at certain things…

Cause and Effect

- Representing Causality and relationships
- System dynamics allow you to represent non-physical relationships too
  - *Morale -> Productivity, Advertising -> Perception,*

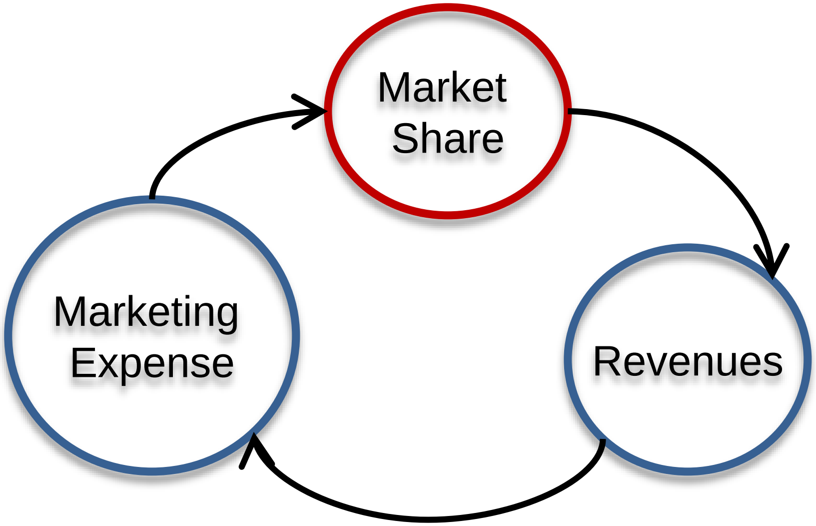
System Dynamics excels at certain things…



Delays

- It takes time for certain effects
- This can be from the nature of the elasticity between variables or from the effects of other mitigating effects
  - *Advertising -> Perception, Actions -> Reputation, Fame -> Perception*

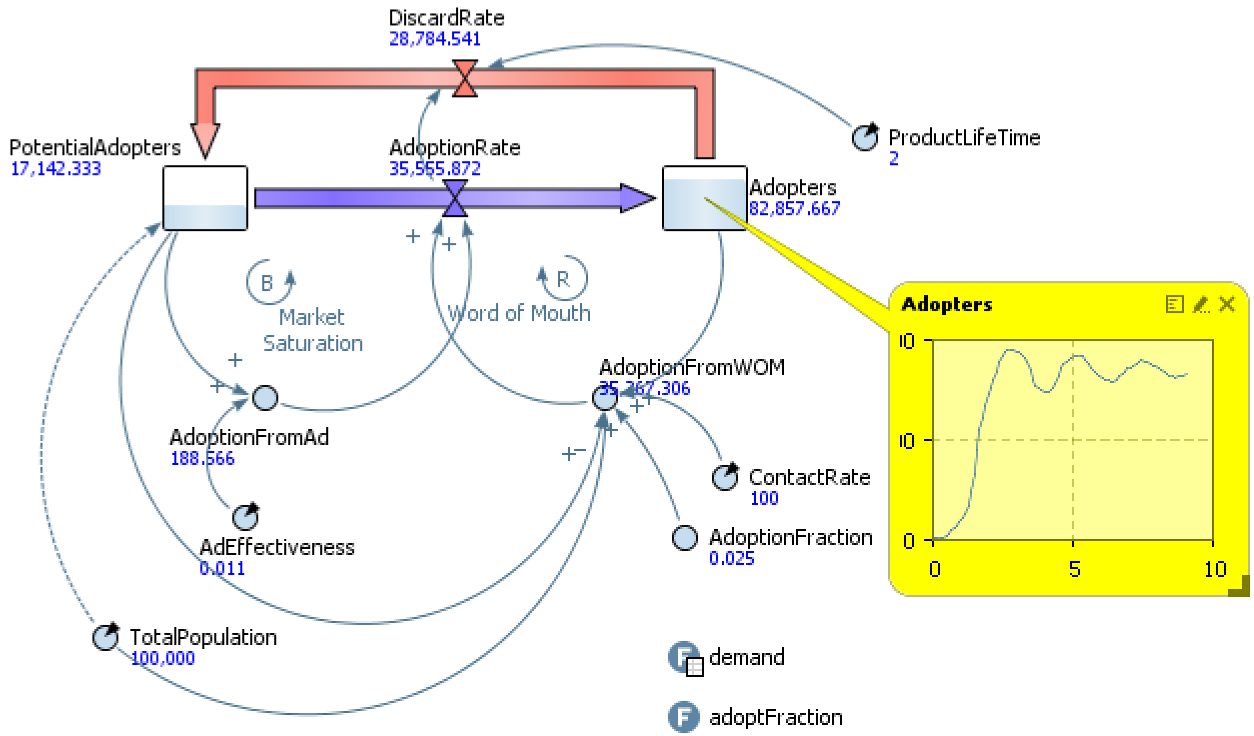
System Dynamics excels at certain things…

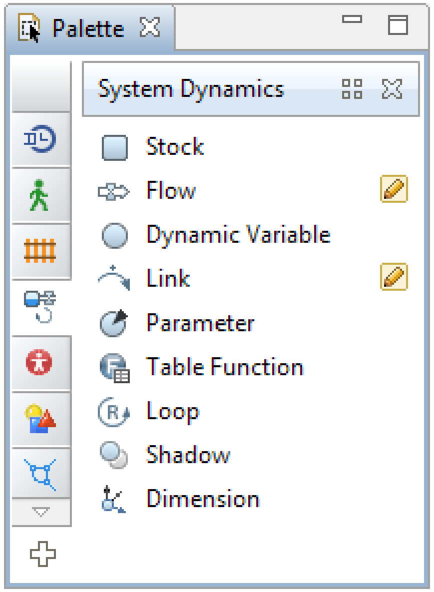


Feedback Effects

- “Practically” unique to System Dynamics, but common in the real world
- A logic error in excel
- Reinforcing Feedback Loops and Correcting Feedback

Stock & flow elements





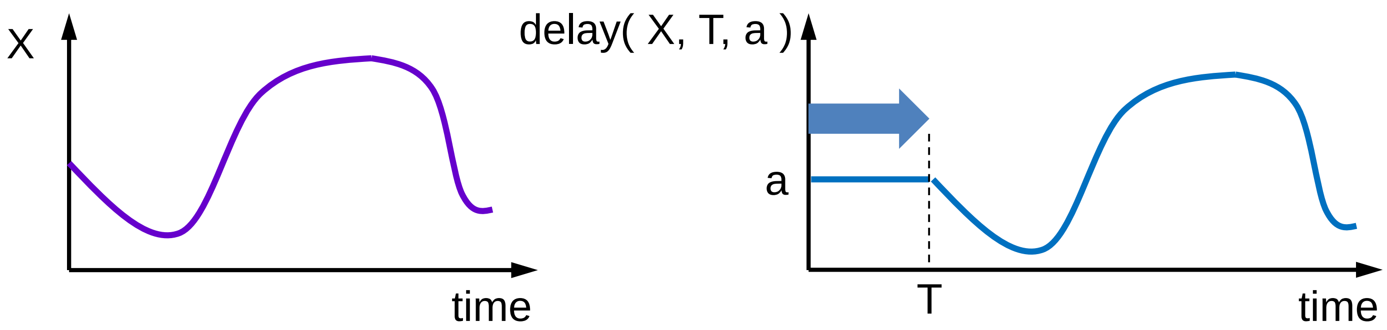
Built-in functions

Mathematical functions

abs(x) cos(x) exp(x) floor(x) limit(min,x,max) log(x) max(a,b) min(a,b) pow(x) round(x) sin(x) sqrt(x) tan(x) …

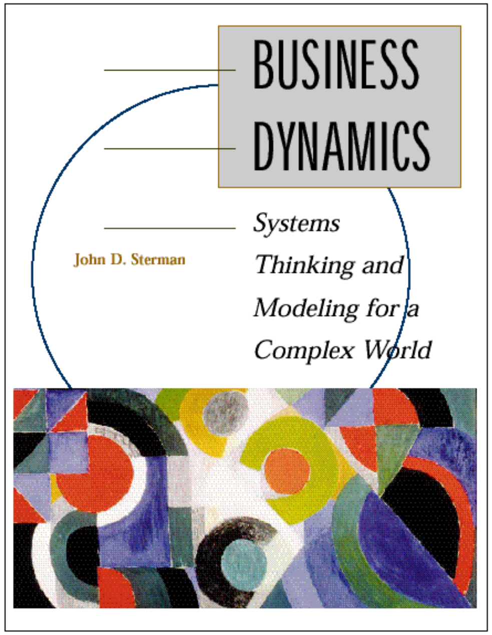
Special system dynamics functions

delay() delay1() delay3() delayInformation() delayMaterial() forecast() npv() npve() smooth() ramp() smooth3() trend() …



See full list of functions in AnyLogic Help -> Advanced Modeling with Java -> AnyLogic functions -> System dynamics functions

The Sacred Book of System Dynamics Modeler

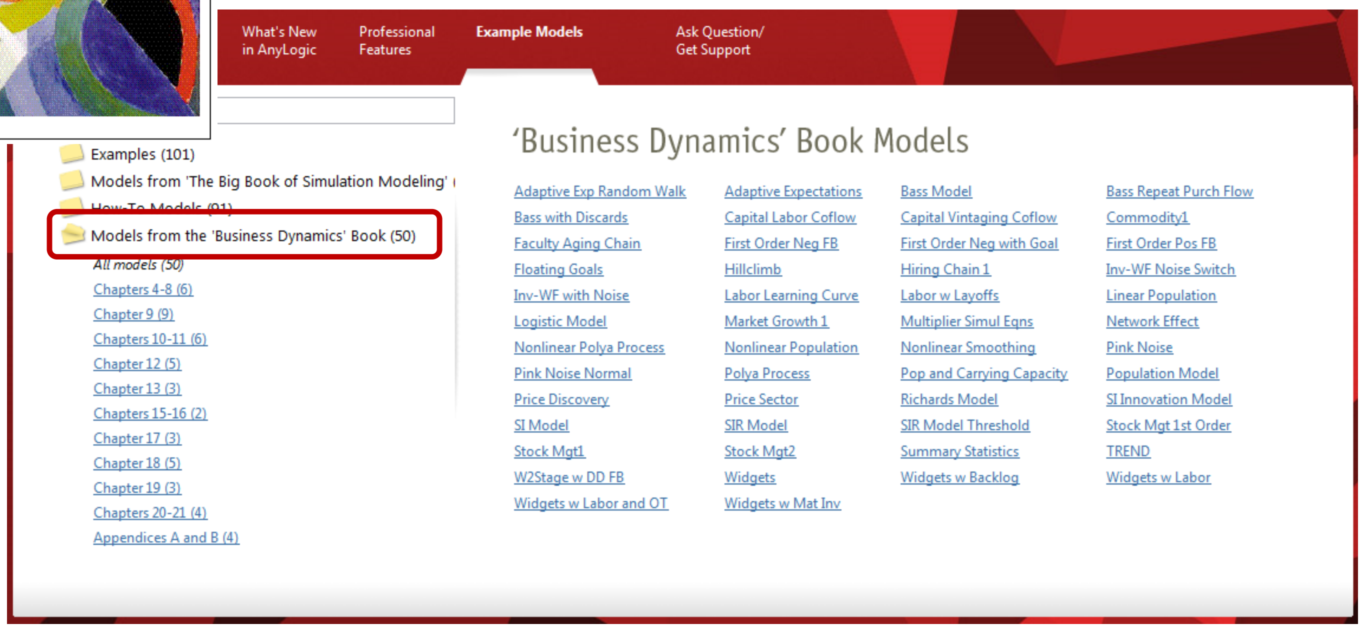


John Sterman

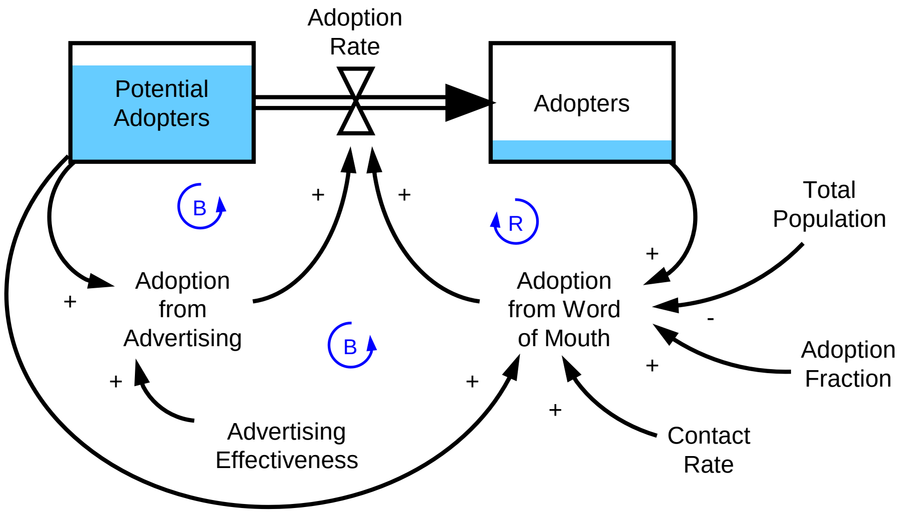
“Business Dynamics:
Systems Thinking and Modeling
for a Complex World”

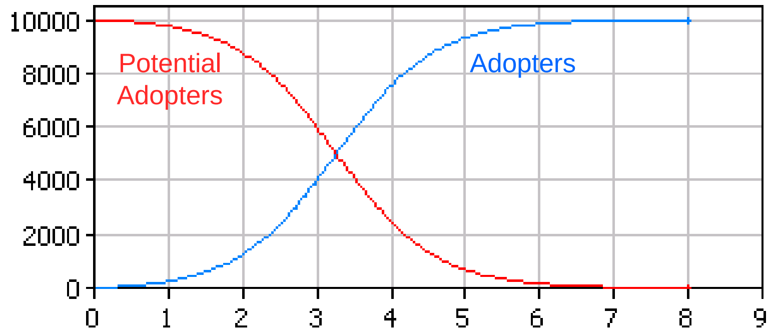
2000. McGraw Hill, 1008 pages

http://www.anylogic.com/business-dynamics-book-models



Bass Diffusion Model





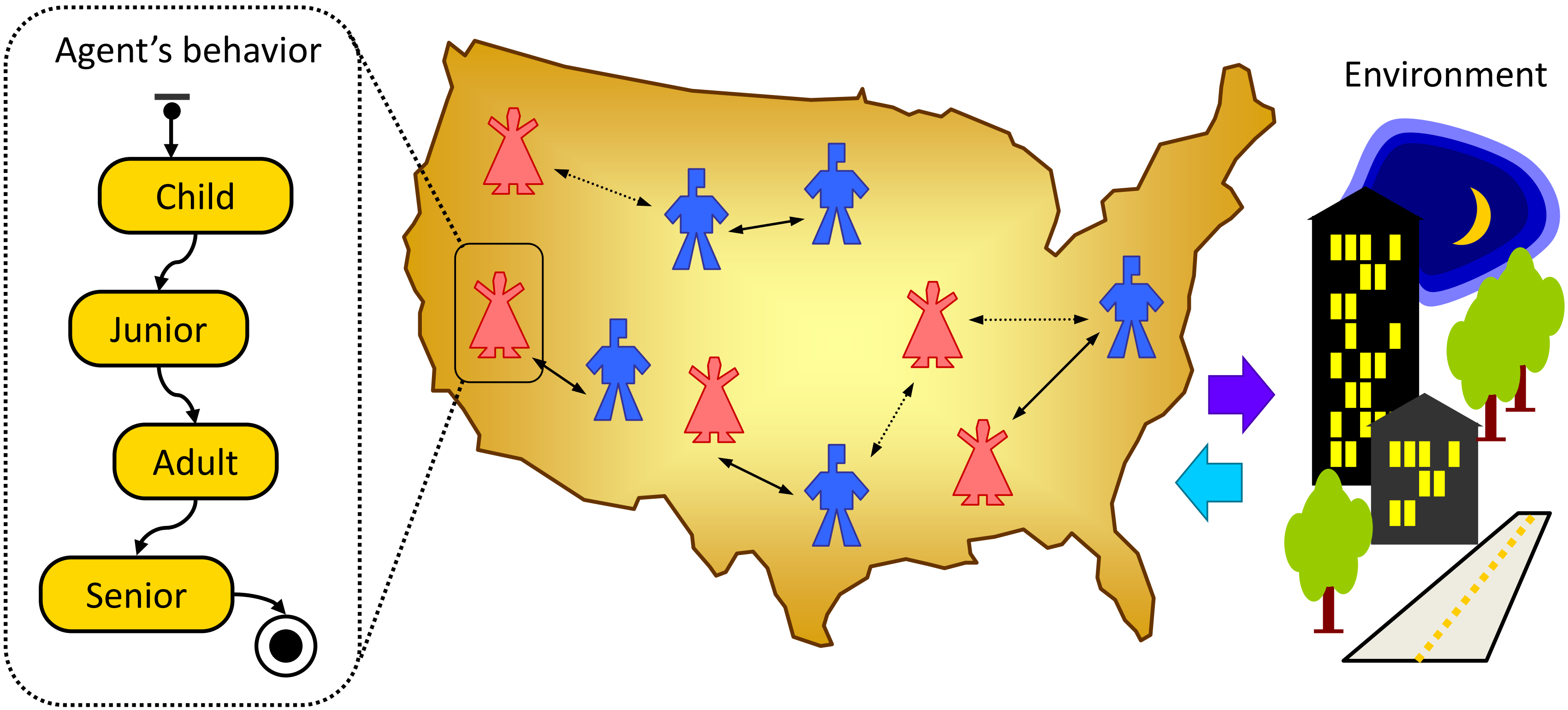
Bass Diffusion

- SD classical textbook model of product, or innovation diffusion
- A population group is considered to consist of Potential Adopters and Adopters; all people behave exactly same way
- Potential Adopters become Adopters at Adoption Rate which depends on advertising
- Advertising goes on all the time, and every time unit it converts Advertising Effectiveness part of Potential Adopters into Adopters
- Initially:
  - *Potential Adopters = 100000*
  - *Adopters = 0*
- Parameter values:
  - *Advertising Effectiveness = 0.011*

Agent Based Modeling

Agent based modeling

- We focus on individual objects and describe their local behavior, local rules
  - Sometimes, we also model the dynamics of the environment



Agents can be:

People:

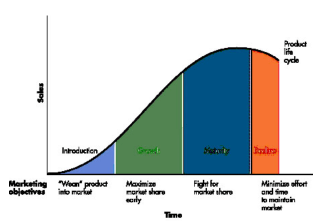
consumers, habitants, employees, patients, doctors, clients, soldiers, …

Vehicles, equipment:

trucks, cars, cranes, aircrafts, rail cars, machines, …

Non-material things:

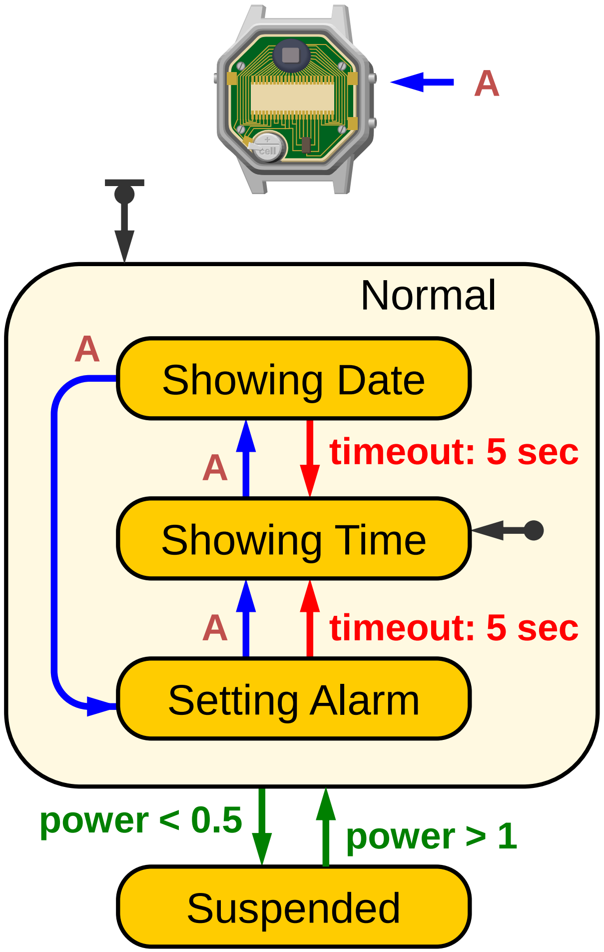
projects, products, innovations, ideas, investments …



Organizations:

companies, political parties, countries, …

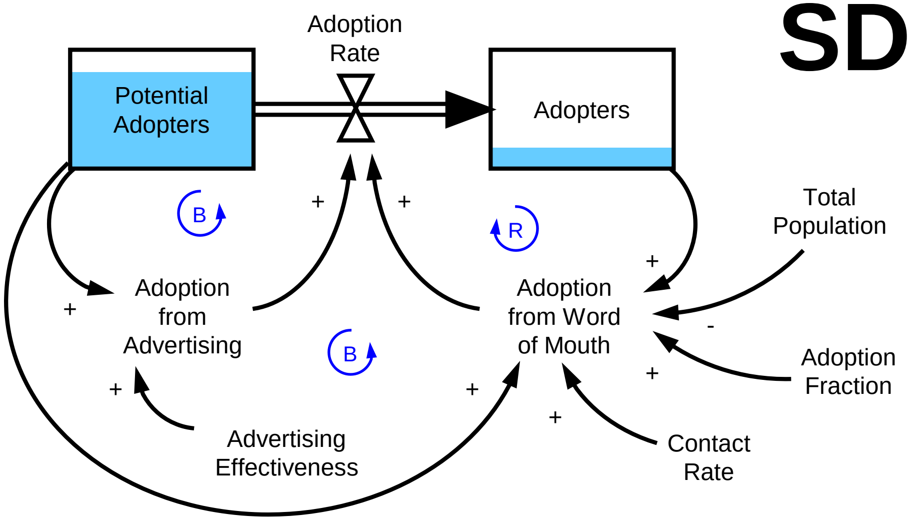
Statecharts



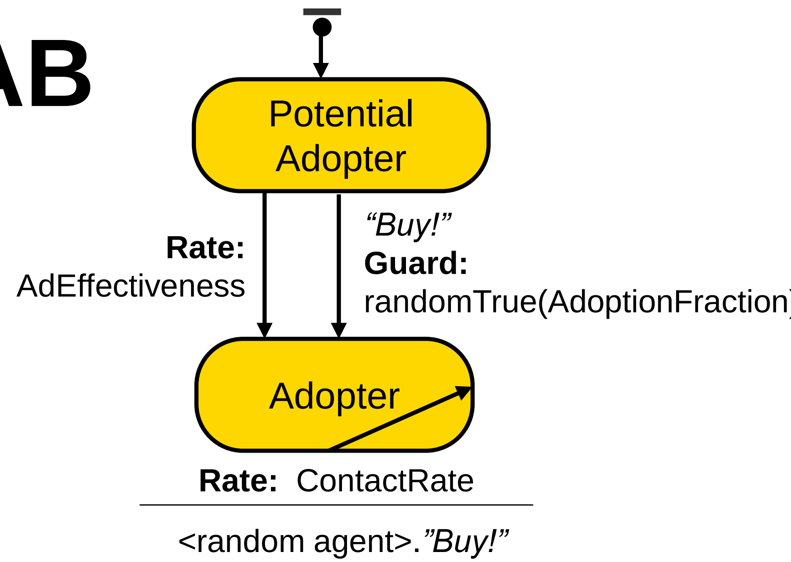
- The most powerful and naturally visual construct
- Statecharts can be used to define:
  - object states / modes of operation
  - response to the external or internal signals and conditions
  - event and time ordering

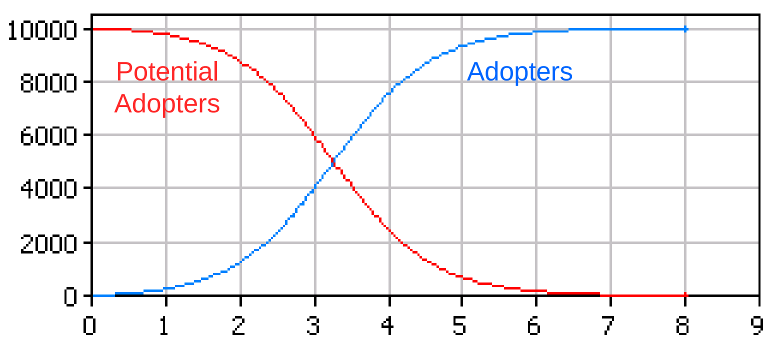
Bass Diffusion – Agent Based version

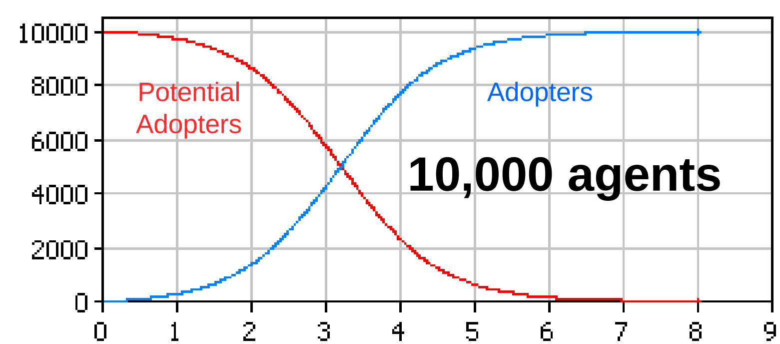
SD



AB

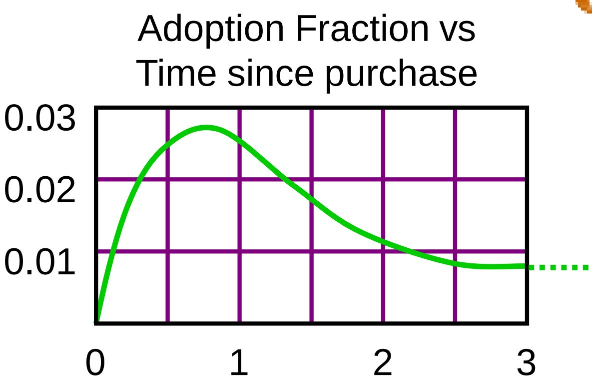


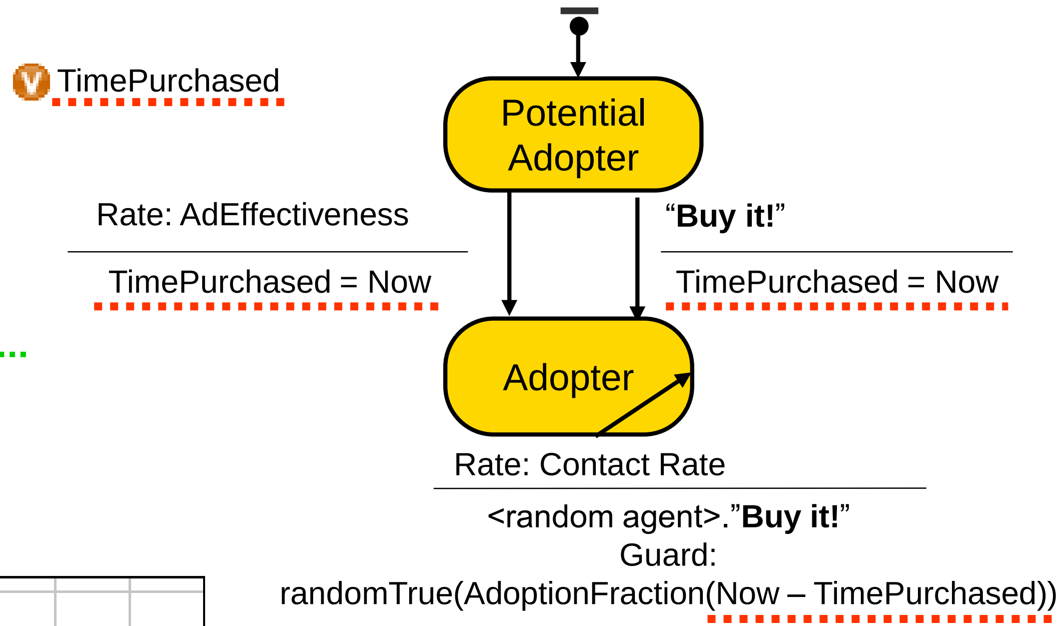


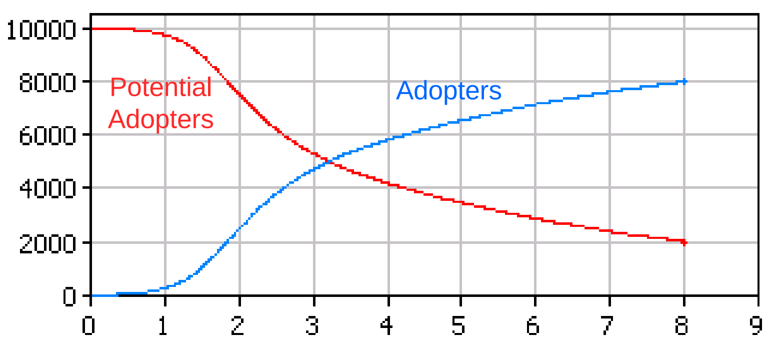


Capturing more with AB Model

- Let the word-of-mouth influence of an adopter depend on how recently he has purchased







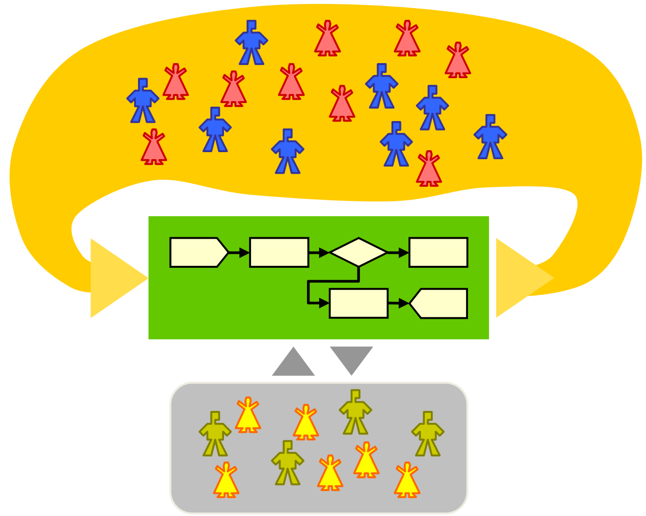
- Can you build an SD model that captures such dynamics?

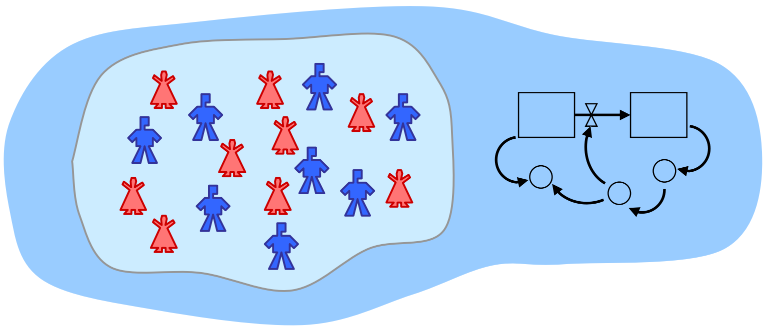
Which approach to use?

- If the problem requirements fit well into the DE or SD modeling paradigms – you can safely use these *traditional approaches*
- In cases where your system contains active objects (people, business units, animals, vehicles, or projects, stocks, products, etc.) with timing, event ordering or other kind of individual, autonomous behavior – *You will benefit from applying the AB approach*
- Sometimes these requirements are at the sub-model level. Then you can consider mixing different approaches in one model and applying most appropriate technique where needed

Multi-paradigm model architectures

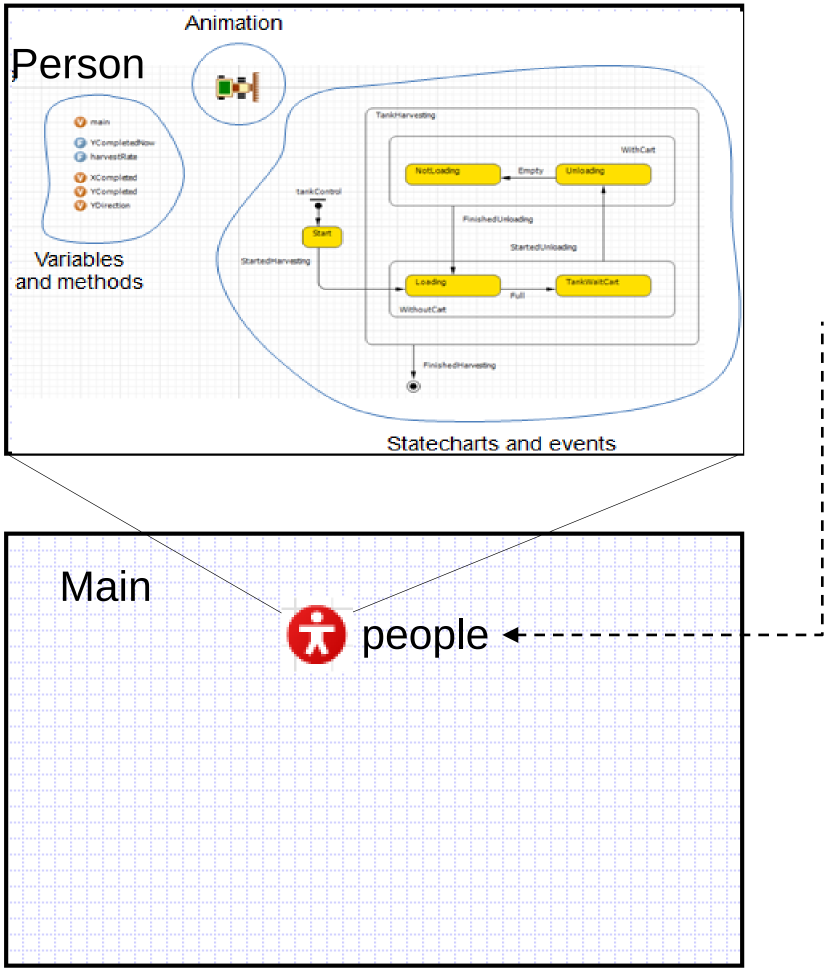
Agents (e.g. customers) interact with other agents (staff) in a Discrete Event flowchart





Agents live in an Environment modeled in System Dynamics way

Typical architecture of an AB model



Name: *people*
Type: *Person*
Replication: *100000*

Adding/removing people:

```
add_people();
remove_people( p );
```

Iterating through all people:

```
for( Person p : people ) {
   …
}
```

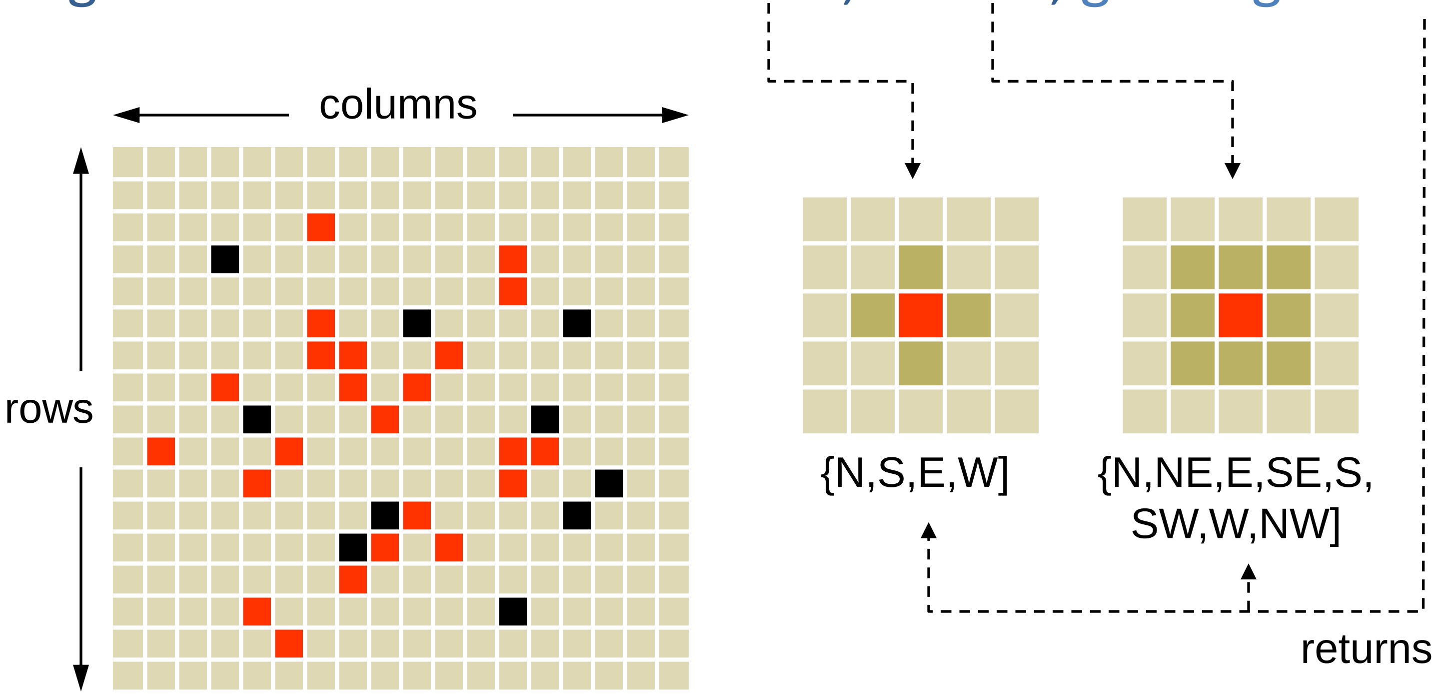
Space: Discrete

- 2D array of cells *Rows* by *Columns*

  At most one agent per cell; to retrieve location: getR(), getC()

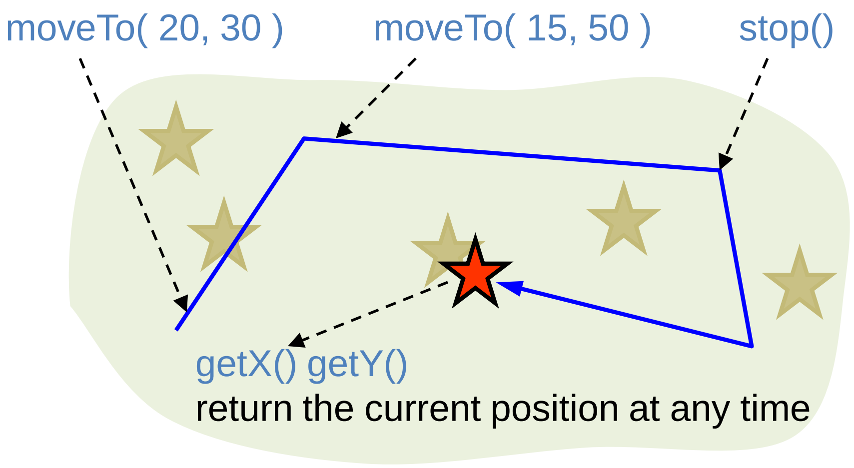
  Movement: jumpToCell(), jumpToRandomCell(), etc.

  Neighborhood models: Euclidean, Moore; getNeighbors()



Space: Continuous

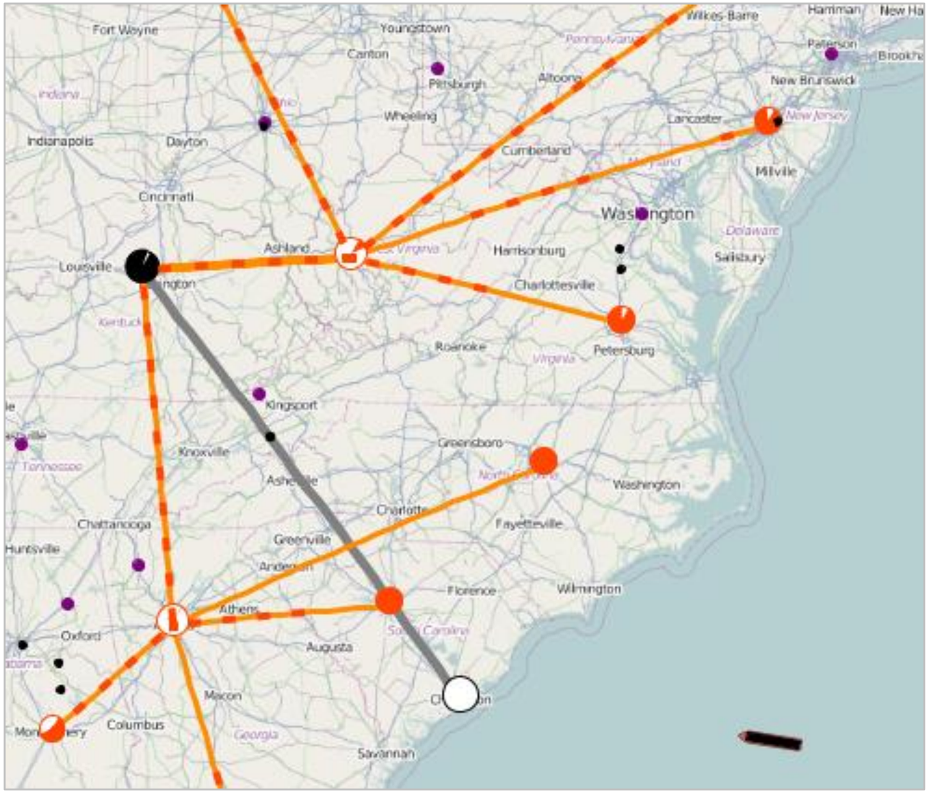
- Agent has (x, y, z) coordinates in 3D space
- Use agent API:
  getX(); getY();
  distanceTo(agent);
  moveTo(x, y, z);
  jumpTo(x, y);
  stop();
  isMoving();
  timeToArrival();
  setSpeed (speed);
- Define action:
  On arrival



Space: GIS

- Agents in a geospatial environment defined by GIS map
- Agent has real (latitude, longitude) coordinates in space
- Use agent functions:

  getLatitude();
  getLongitude();
  distanceTo(agent);
  jumpTo(x, y);
  moveTo(x, y);
  stop();
  isMoving();
  timeToArrival();
  setSpeed (speed);

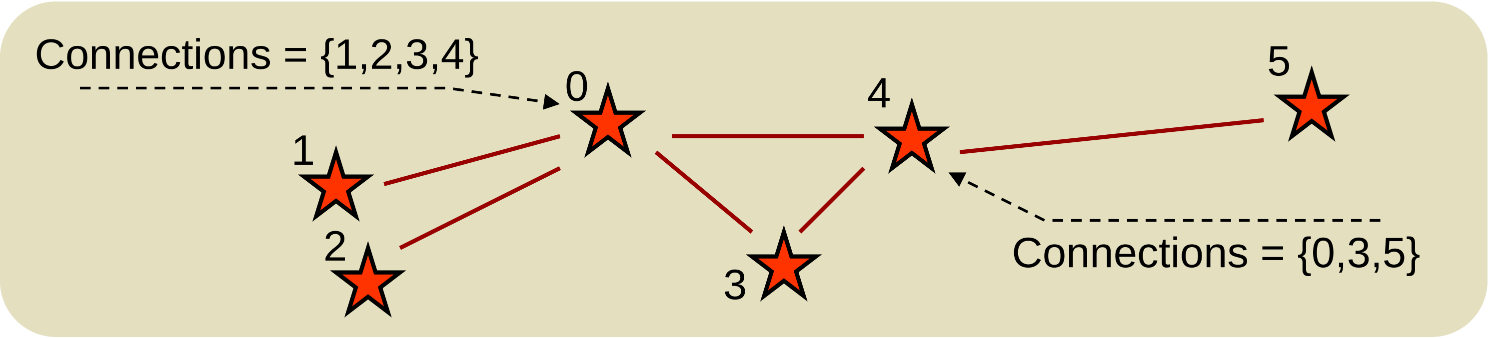


- Define action:

  On arrival

Network: connections and communication

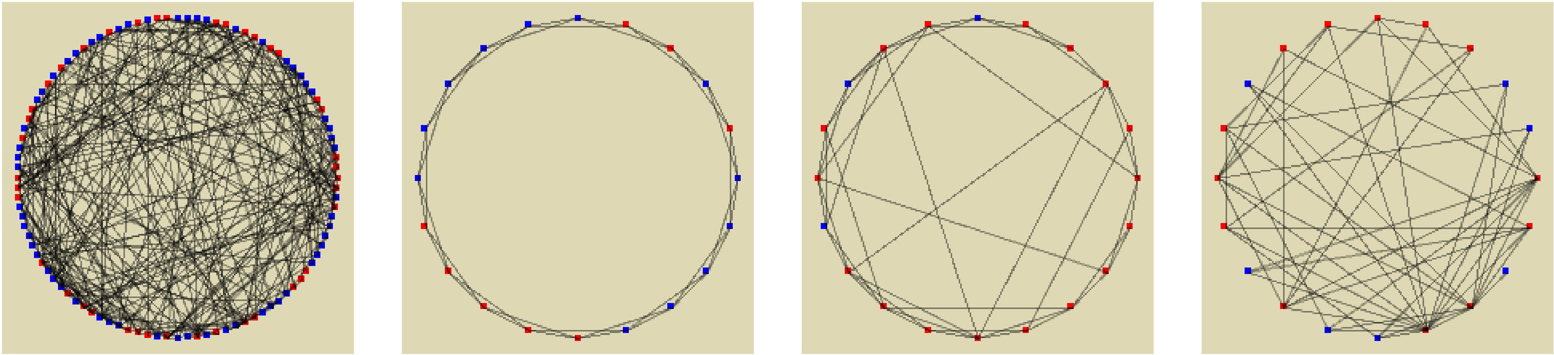
- Every agent has a list of connections – other agents
  - Use standard network types or define your own using API:
    connectTo(agent); disconnectFrom(agent);
  - Access the collection of connected agents:
    getConnections(); getConnectedAgent(i);
  - Communication in network:
    Send messages:
    sendToAll(msg); sendToRandom(msg); sendToAllConnected(msg); sendToRandomConnected(msg); send(msg, agent)
    Define reaction in *connections* element: On message received



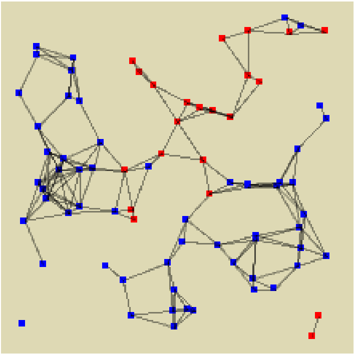
Network: Standard types

- Standard network types:

  Random, Ring lattice, Small world, Scale free,



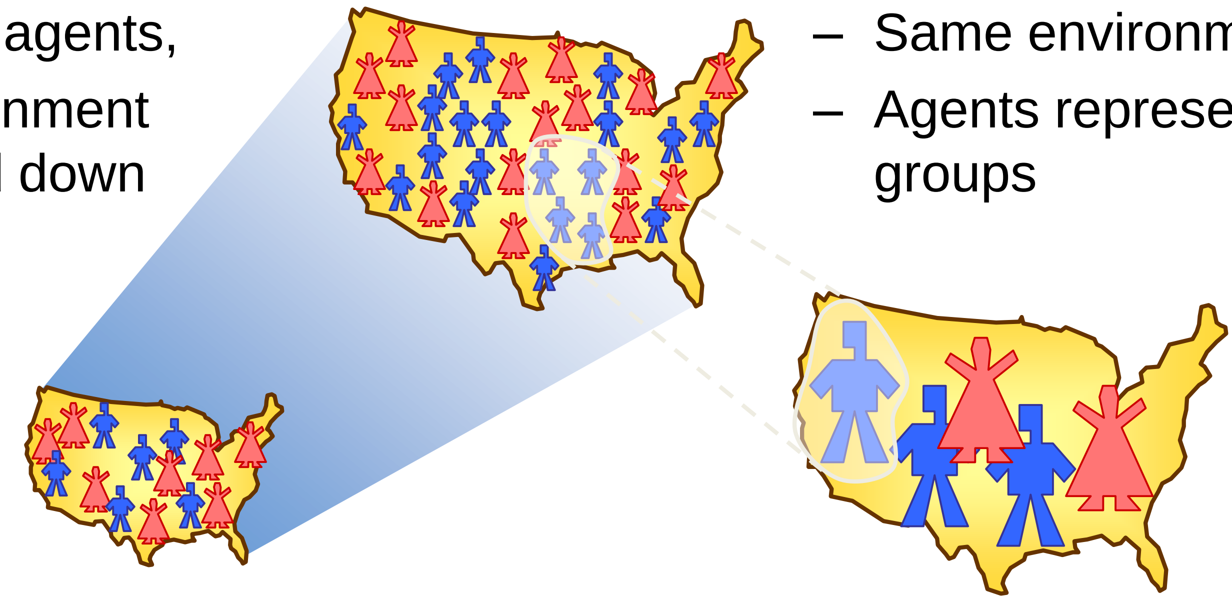
  Distance based (layout dependent)



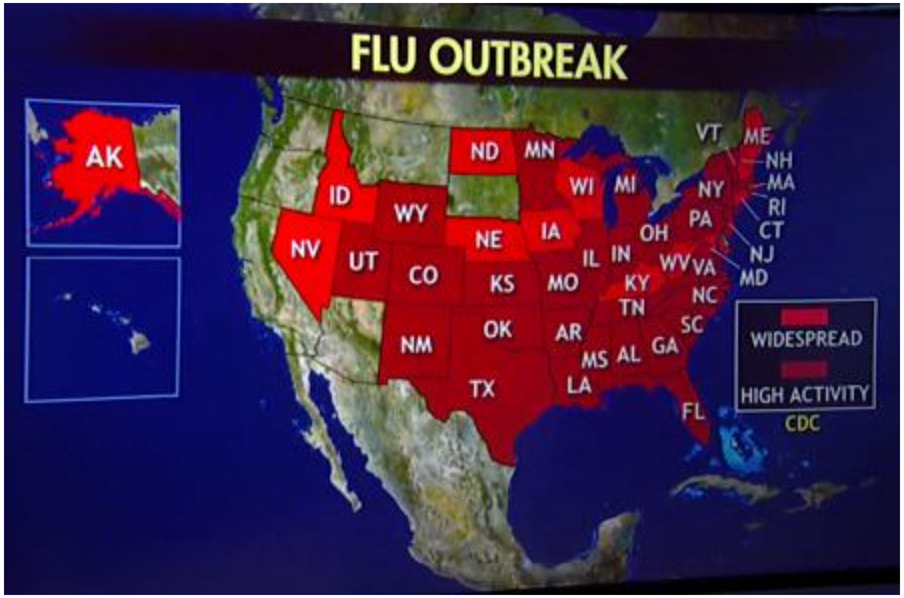
- You can
  - combine standard and custom networks
  - re-apply standard network during run, etc.

How many agents to simulate?

- If I need to model the US population do I need to simulate 300,000,000 agents? **Fortunately not!**
- Two main "model scaling" techniques are used:
  - Same agents,
  - Environment scaled down
  - Same environment
  - Agents represent groups



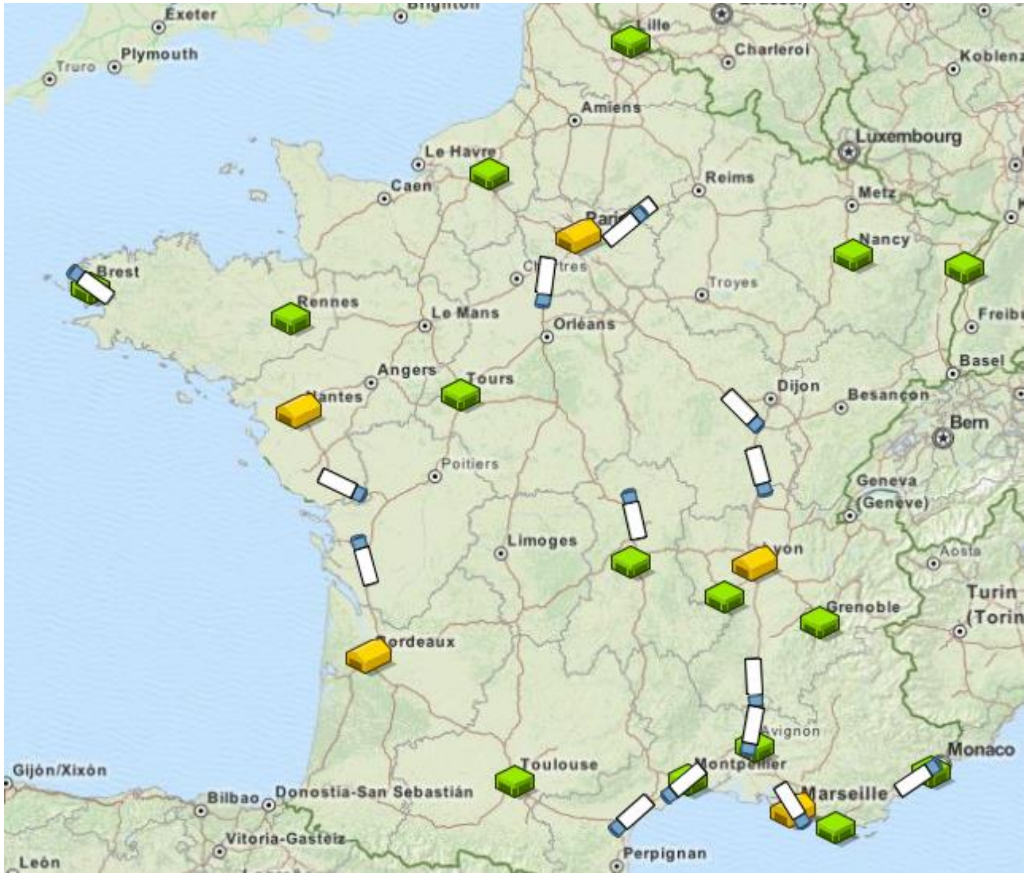
Disease Spread Model



Disease Spread Model

- In the model each person has 3 possible states: Susceptible, Infectious or Recovered (SIR). Initially all but a few people are susceptible, and a few are infectious. Upon contact with an infectious person a susceptible person will get the disease based on a certain probability.
- The agents are placed in a continuous space. The contacts occur between random agents.

Supply Chain Model



Pedestrian Modeling

What are pedestrian models built for?

- At a preliminary project assessment stage
  - Assess the ability of a facility to cope with a planned loading and comply with safety requirements
- At the stage of the design of a new facility
  - Assess alternatives, promptly assess revisions, seek the best solutions
- During construction /maintenance works at an operating facility
  - Seek the least inconvenient temporary routes
- At operating facility
  - Increase a throughput capacity, arrange queues
  - Optimize the operation of services (number of personnel, working hours)
  - Allocate signage
  - Assess the throughput capacity of a facility at a planned increase of loading
  - Optimize time schedules (for example, train schedules)
  - Allocate advertisement, goods, retail outlets
- Safety
  - Plan escape routes
  - Vulnerability assessment for terroristic attacks and catastrophes
- As well as for presenting your project in a contest
  - Pedestrian models enable to obtain high quality and convincing animation and vividly demonstrate your offer

Which facilities are modeled?

transport

- Railway stations
- Metro stations
- Airports
- Pedestrian passageways

"attractions"

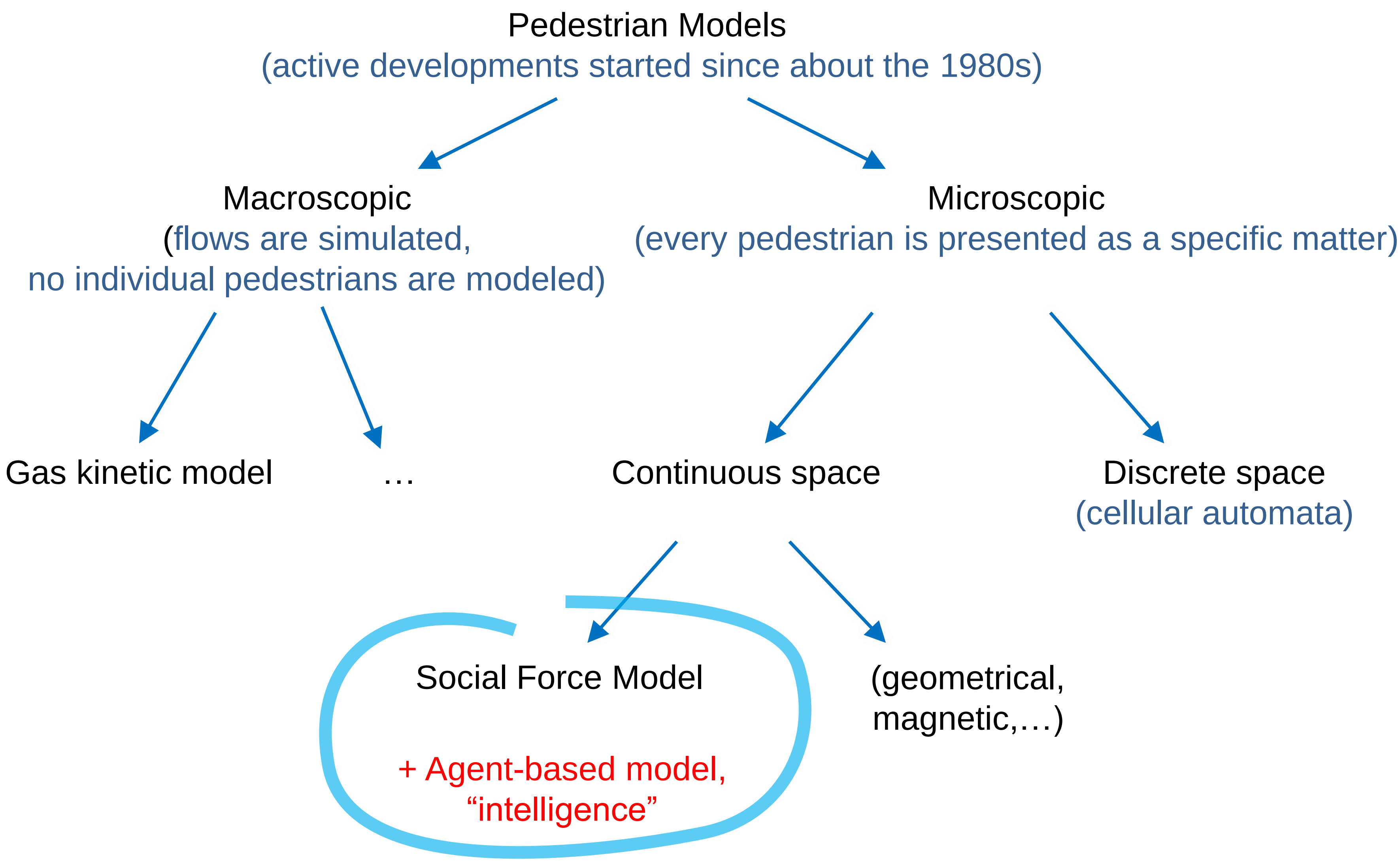
- Shopping malls
- Museums
- Amusement parks

events

- Stadiums
- Concert halls
- Street events (festivals, rallies, demonstrations)

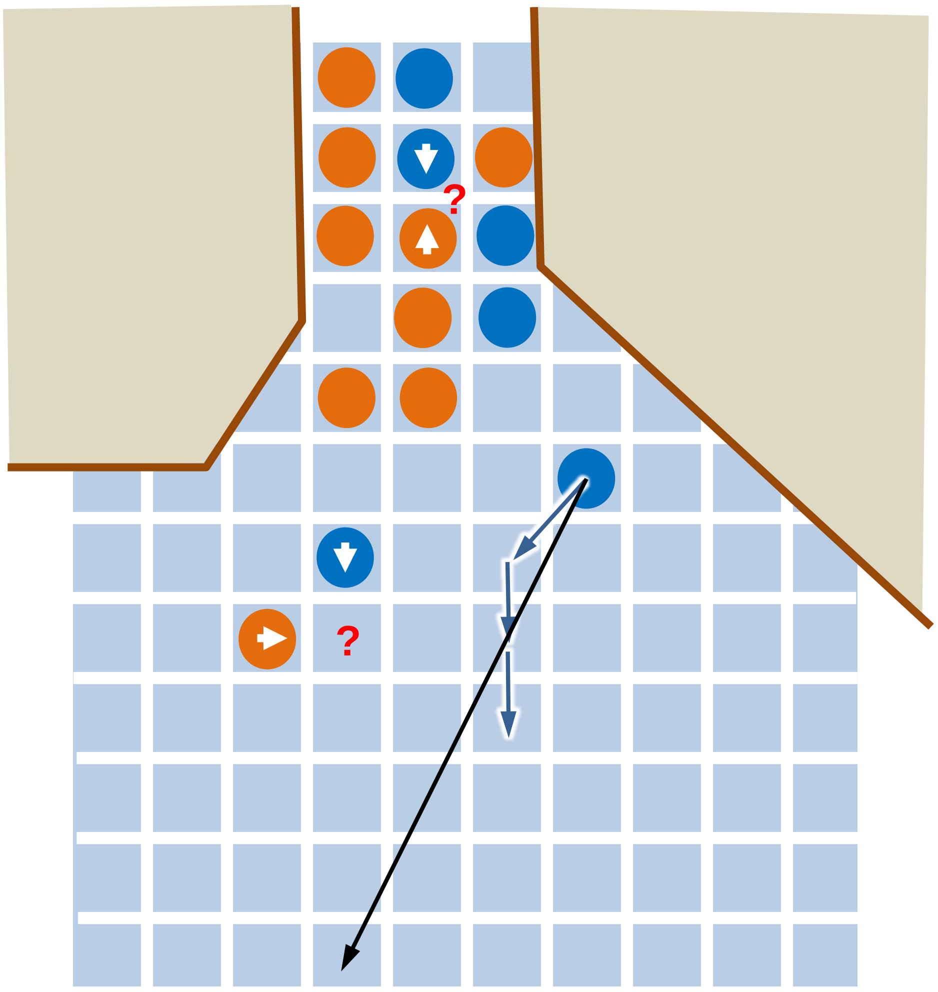
In general all the facilities where the arrangement of physical space for pedestrians affects throughput capacity, quality of service, and safety

Theory. Pedestrian Model Types



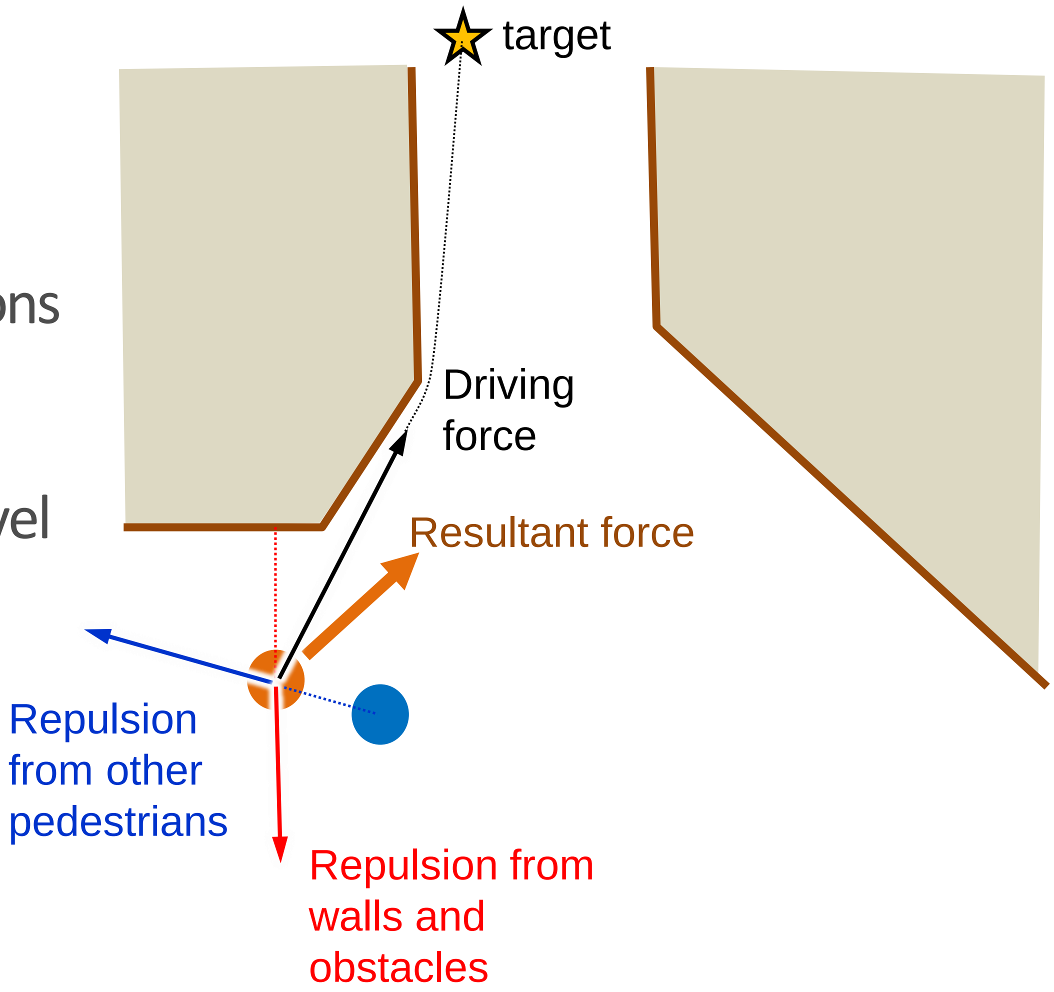
Cellular automata

- Easy local rules
- Fast-to-calculate
- Can be well-calibrated
- Poor animation
- See Blue & Adler

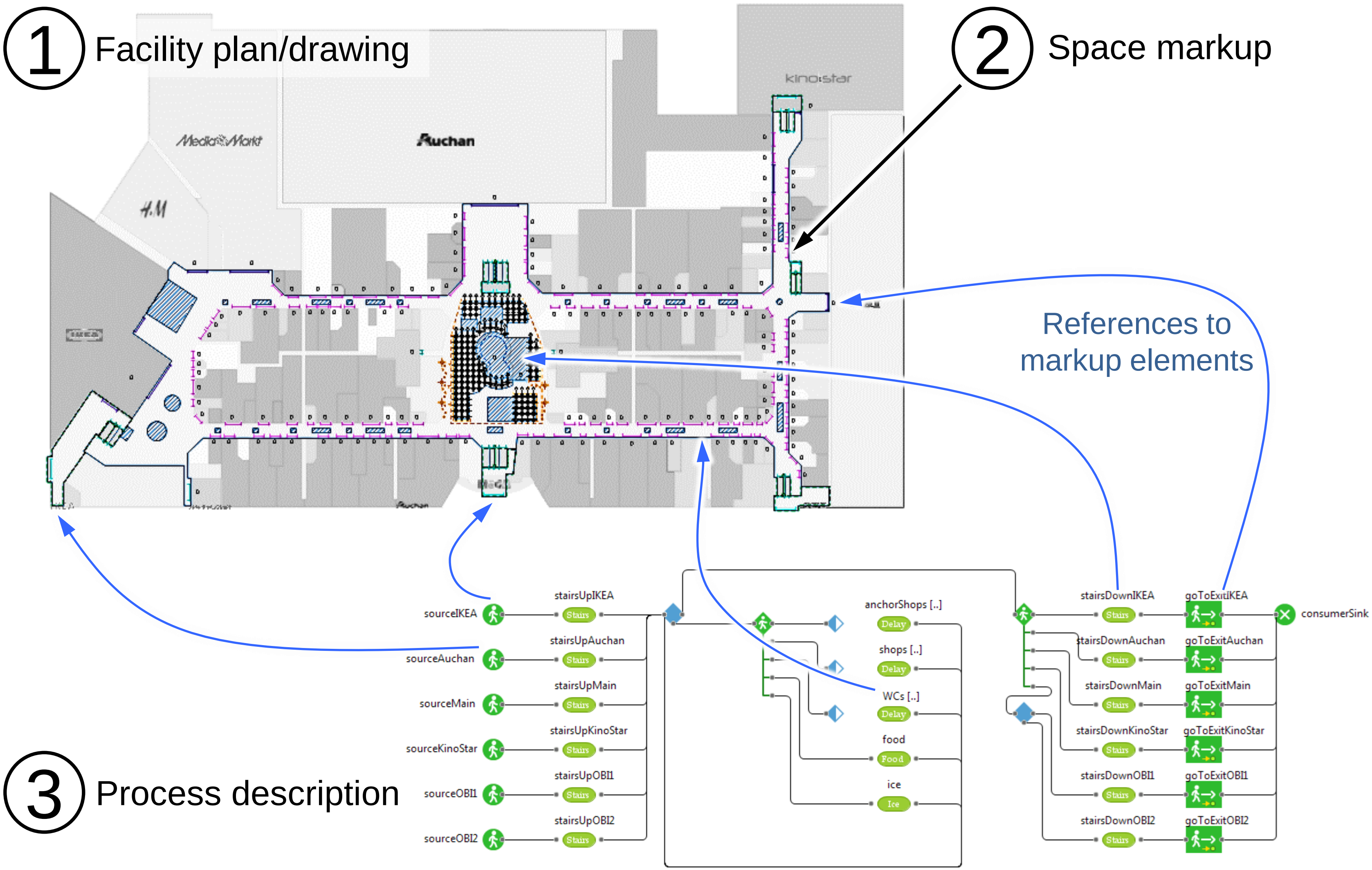


Social Force Model

- Newton mechanics
- Realism
- Relatively slow calculations
- Very realistic animation
- Extended with higher level decision making logic
- See Helbing & Molnar

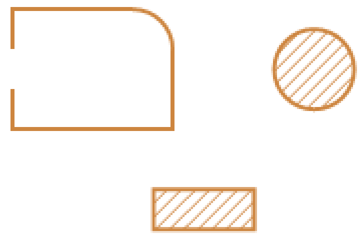


How are pedestrian models built?

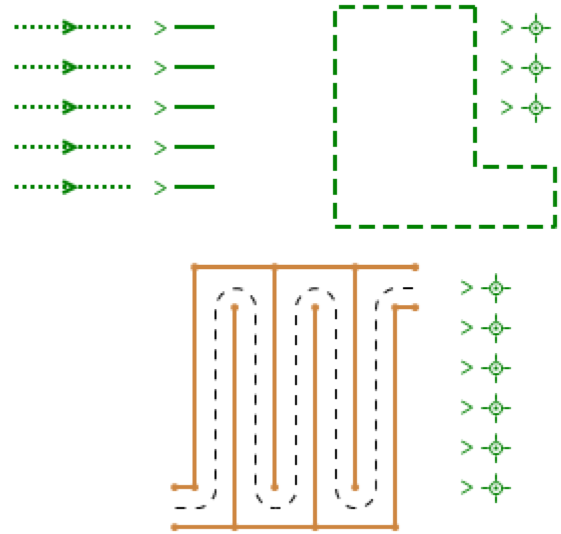


Space Markup elements

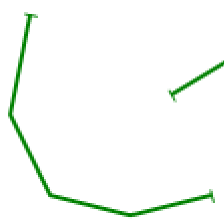
Walls



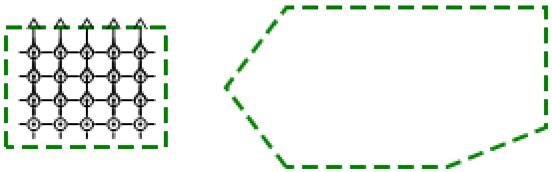
Services (service points) and queues



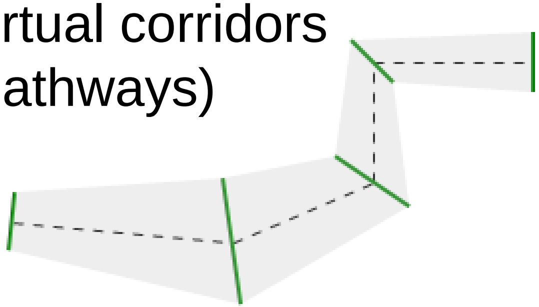
Target lines / pedestrian appearance lines



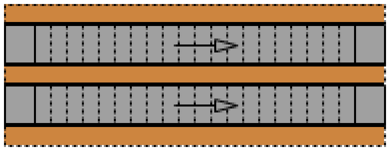
Waiting areas / target areas



Virtual corridors (pathways)



Escalator

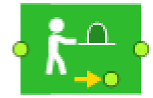


Process Description Basic Blocks

PedSource

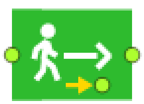
Creates pedestrians on a line, at a point or in an area with a given rate, according to a time schedule, etc.

PedService



Sets servicing parameters (where is a delay, the selection of a queue, etc.)

PedGoTo

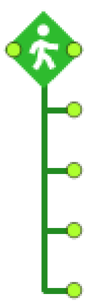


Sets up an objective or a route

PedWait

Sets waiting parameters (where to wait, in relation to time, until an event)

PedSelectOutput



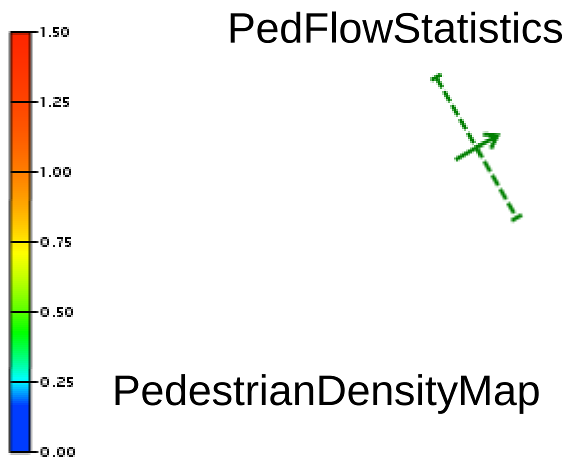
Divides a passenger flow

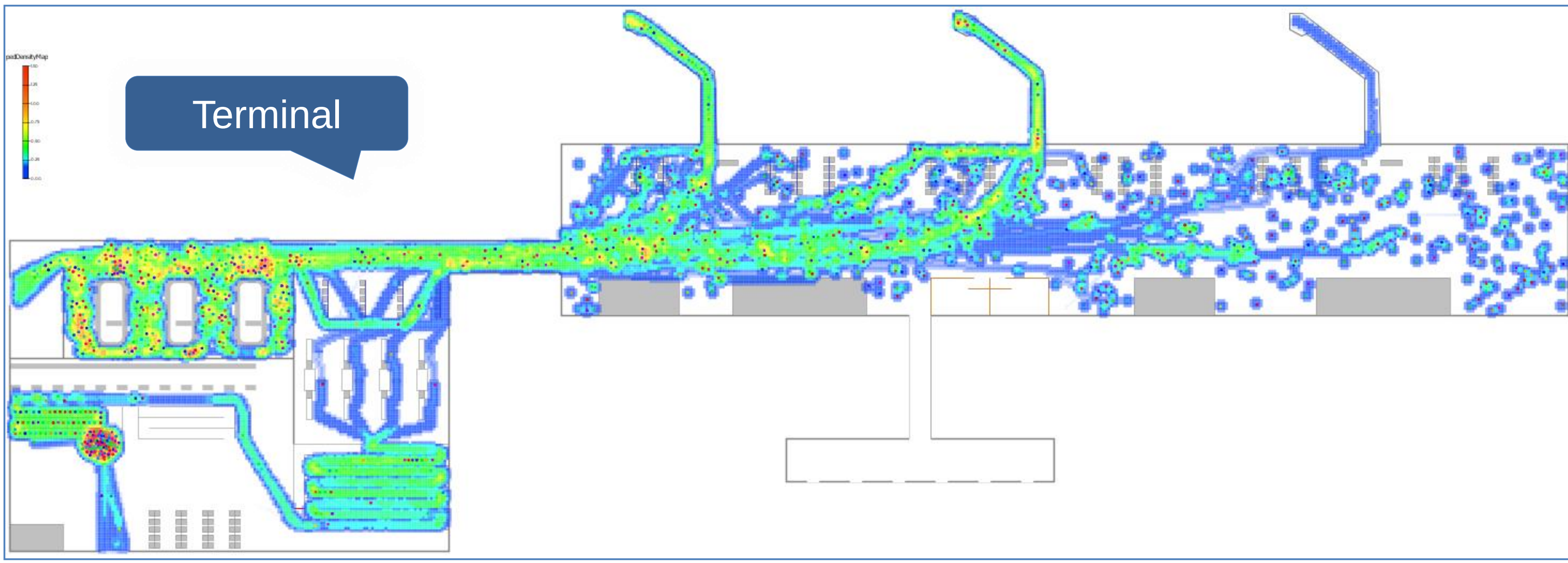
PedSink

Deletes passengers from the model

Measurements and Statistics in Pedestrian Models

- Metrics specific for pedestrian models
  - Flow characteristics: the total number of passenger having passed through a section per a unit of time,
    the same quantity per a unit of length
  - Density in a certain area:
    the number of passengers per square meter (average per a unit of time);
    density charts





Individual Features of a Pedestrian

- Since each pedestrian is modeled as an agent, individual features can be adhered to them
- Features built in a basic model:
  - Comfortable speed
  - Dimension (“diameter”)
- These can be added to them:
  - Individual targets (flight, platform, shop)
  - Servicing class (first / business / economy)
  - Citizenship (US/EU/other, …)
  - Servicing speed
  - …
- These features can be checked during the pathway of a pedestrian throughout a process diagram and affect their behavior

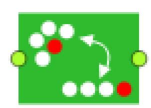
Groups

- Pedestrian behavior in groups considerably differs from that of independent pedestrians
  - Hold together; how to go ("in a rank", "in a convoy", "in a flock")?
  - The presence or absence of a leader (for example, a guide)
  - Service: one pedestrian is serviced for all? (buys tickets for all, whereas a security check should be passed by everybody)
  - Does everybody stand in a queue? Where are those who are not standing in a queue waiting?

PedSource

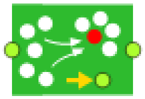
Is able to set groups, sets initial construction as well as default behavior at servicing points

PedGroupChangeFormation



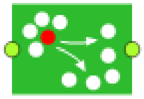
Changes a group formation type

PedGroupAssemble



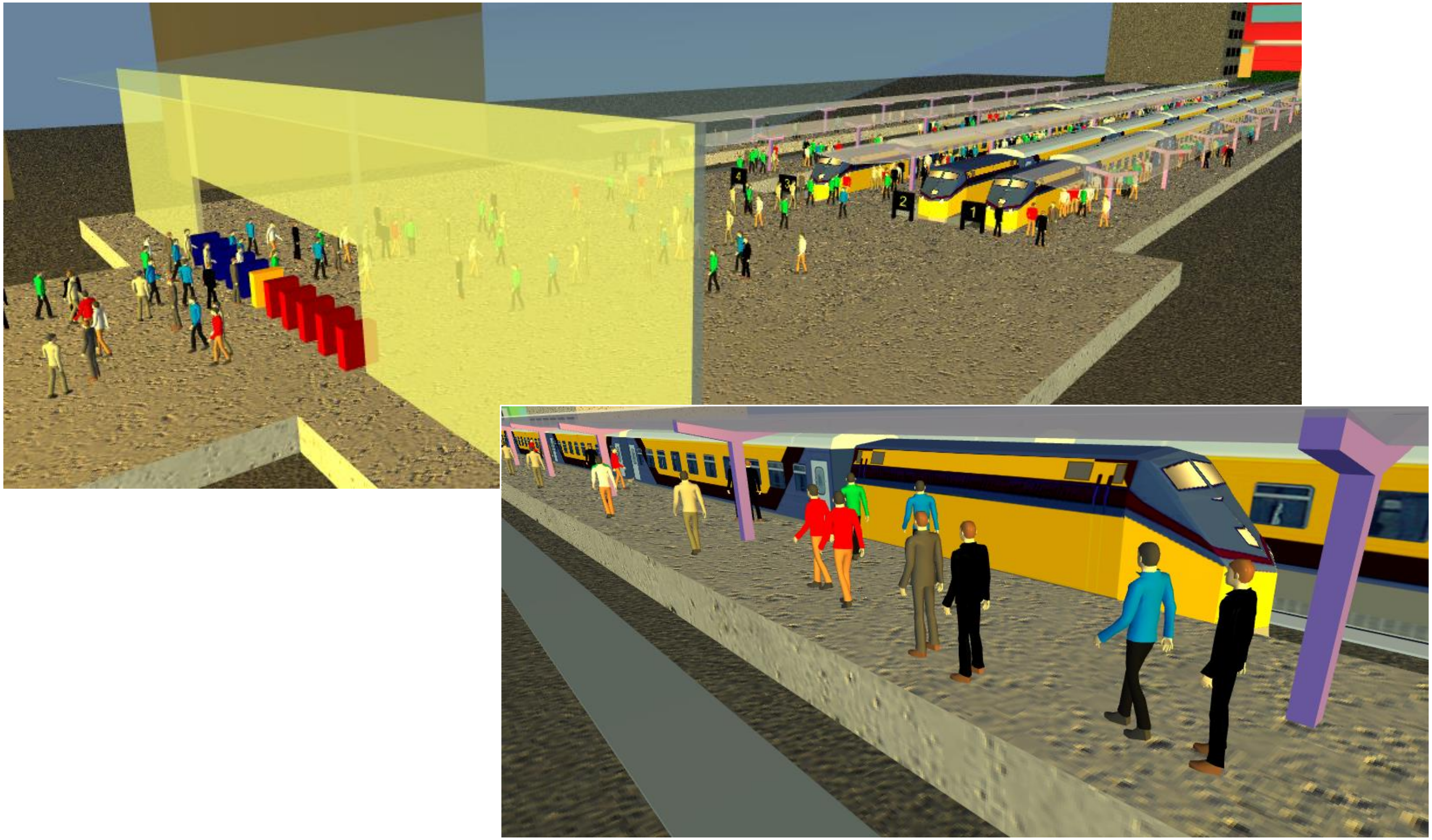
The same but out of the existing independent pedestrians

PedGroupDisassemble



Divides a group into individual independent pedestrians

Railway Station Model



Group Exercise

Instructions

1. Break into teams and select a team name
2. Review the details and performance metrics of the existing process
3. From the list of improvement options, select the ones you believe will be the most impactful. Each option has a cost and the team has a budget
4. Reconvene and a simulation model will be used to show the outcome for each team's selections

Further Information

Steps For a Successful Project

- Follow a structured methodology
- Formalize the team
- Select the software
- Commit to the project
- Define success
- Consider key success factors
- Keep up with the industry

Steps: Structured simulation methodology

- Define the problem or objective
- Develop a formal Simulation Specification
- Model the real or proposed system
- Gather model data
  - Historical
  - Observation / time studies
  - Estimates
- Verify and validate model
- Run "what-ifs" & analyze results
- Implement the changes!

Steps: Define the objectives

- Communicate the written objectives
- Appreciate the unwritten objectives
  - Upfront and unbiased planning
  - Other methods have failed
  - Validate someone's ideas
  - Justify something already done
  - Provide a check and balance
  - Try out a gee-whiz tool
  - Visualize the process
  - Develop a training tool
  - Build a baseline for future use

Steps: Develop a Simulation Specification

- Objective and scope
- System Description
- Assumptions
- Deliverables
- Schedule
- Structure (approach, tool(s), animation)
- Interface, inputs, outputs
- Data collection strategy
- Validation methods
- Resource requirements
- Planned scenarios

Steps: Model validation (ER example)

- Compare actual versus simulation
  - Correlation study
- Comparison to direct observation. For emergency department, for example:
  - Arrivals (numbers, day, time)
  - Patients in waiting room
  - Patients in hallway
- Subject matter expert review
- Demo of model animation for the users
  - Queues, paths, bottlenecks

Steps: Formalize the team

- Project manager
- Process designers
- Process owners
- Process users
- Data sources
- Upper management
- Simulationists
- Simulation users

Steps: Today's simulation tools

Easier to user interface

Industry-specific custom templates

Improved model build and run time

Used throughout system lifecycle

Numerous simulation applications

Steps: Tool considerations

- User interface
- Flexibility
- Upgrade progression
- Data exchange
- Animation (none, 2D, 3D)
- Run-time license or viewer
- Object-oriented design
- Custom object and template development
- Continuous and/or discrete
- Optimization
- Customer care
- Market breadth
- Price

Steps: Commit to the project

- Allocate $ for software purchase and maintenance/support
  - Initial software purchase ($1-25K)
  - Annual software maintenance
    - *Higher cost software - % of purchase*
    - *Lower cost software – purchase upgrade*
- Attend offsite or uninterrupted training
  - General appreciation-level
  - Detailed simulationist-level
- Identify the organization and person(s) responsible for simulation in the company
- Conduct a small pilot project
- Provide visible management support
- Maintain proficiency

Steps: Define success

- Tangibles
  - Results that drive decisions
  - Verified improvement
- Intangibles
  - Process understanding
  - Team building
  - Employee involvement
  - Management visibility
  - Success story

Steps: Consider key success factors

- Start small & with a pilot project
- Build ownership of the model & results
- Bound the project through defined scope
- Use a hierarchical modeling methodology
- Spend enough time developing and communicating assumptions
- Get the help you need
- Target the lowest fidelity level possible
- Ensure management visibility
- Maintain momentum

Steps: More key success factors

- Involve as many as possible as early as possible
- Take pictures or movies of the actual system and use them in discussions
- Collect more than process step data
- Develop descriptive (yet clear) graphics
- Validate the model with all constituents
- Consider the downstream user
  - Interface tailored to user
  - Reuse
  - Documentation
- Develop the “believable baseline”

Steps: Keep up with the industry

- Societies
  - Institute of Industrial & Systems Engineers (www.iise.org)
  - The Society for Modeling & Simulation International (www.scs.org)
  - Institute for Operations Research and the Management Sciences (www.informs.org)
- Conferences
  - The Winter Simulation Conference
  - INFORMS Annual and Analytics Conferences
  - Some vendors have user conferences
- Books & Periodicals
  - The Big Book of Simulation Modeling
  - AnyLogic in 3 Days
  - Conference Proceedings
- Search the web!