STEEL MANUFACTURING MODEL

Process Modeling and Simulation

Steel Manufacturing

Professor. Dr. Yilmaz Uygun Group 2: Hala Abuhassan, Alexandra Gkragkopoulou, Joelle Karadsheh, Nada Martinovic December 3, 2023

Introduction (Alexandra)

The goal of this assignment is to simulate the manufacturing of steel. Steel is mainly used while constructing buildings, infrastructure, tools, ships, automobiles, machines, appliances and weapons. The main steps that are involved in the manufacturing of steel are: melting of scrap, degassing, compact strip production, cold strip milling. Our report consists of a 2D animation, graphs and diagrams for total production, slab processing time, number of slabs and coils produced, and a graph showing different products. We also discuss important parameters of the system, we run an optimization experiment where we change the input rate, a maintenance model and develop a system dynamics model.

Question 1 (Joelle and Nada)

The model below shows the steel manufacturing plant. Steel is manufactured through different processes. The steel plant produces galvanized and galvannealed sheets, cold rolled fully processed sheets, hot rolled plates, and cold rolled full hard sheets).



Fig.1 Steel Plant



Fig.2 Production flow

Question 2 (Hala)

First graph: Total production

In order to visualize the total production at the steel plant we added a time plot graph and named it "Total Production". We used a time plot specifically to show us the total production over time. It is very important to note that the total production also considers the amount of slab produced in earlier stages as well as coil production, hence our total production consists of (steel slabs, Coil, hot rolled HR, cold rolled hard sheet, GIGA, CRFP, HRP, CRFH). Total production = amount produced * LadlesWeight

The amount of each product produced is taken from the output. Hence, in the time plot and under "value" we add up all the outputs for all the products and then we multiply them with the ladles weight which is in tons. The graph on the right shows an increasing graph because as more time passes we are producing more products.

Properties 않		8	
🖾 plot - Time Plot			
Name: plot Ignore Visible on upper agent Lock			^
▼ Data			
● Value ○ Data set			
Title: Total Production			
Value: (outputHR + outputHRP+ outputCRFH+ outputCRFP+ outputGIGA)*ladlesWei	ght		
Point style: V			
Line width: V 1 pt			
Color: vellowGreen v			
The second se			
▼ Data update			

Fig.3 Time plot properties



Fig.4 Amount produced in tons over time

🔲 Properties 🛛		8	
🌆 chart - Hist	togram		
Name: char	t ☐ Ignore ☑ Visible on upper agent	t 🗌	Lo
Title: Histogram:	Slab processing time timeMeasureEnd.distribution		
PDF color:	green V CDF color:	\vee	м
CDF & mean	line Width: 📃 🗸 1		
Low % color:	♥ High % color:		1
🖶 Add histogi	am data		_
 Data update Appearance 			



Fig.5 Histogram Properties



Third graph: Number of slabs produced

First, we create a bar chart and assign it to the name "Number of slabs produced" Then we set the value in the data to "LadleToSlab,count()" This will count the slabs at the sink because that is the total amount of steel slabs produced. The number of steel slabs produced is 15,373 and that is the amount reached when the max number of agents is reached.





Fig .7 Bar chart properties



Fourth graph: Number of coils produced

we add another bar chart. This time we want to count the number of coils produced. Hence, we consider the sink "SlabToCoil" where the total amount of coil produced is collected. We use a count function to measure the total number of coils produced in this steel plant. We notice that the total amount is 15,364.

La riopenes as	0		
և chart1 - Bar Chart			
Name: chart1	Ignore Visible on upper agent		
Lock			
Scale: ● Auto ○ Fixed ○ 1	100%		
From: 0	To: 1		
Update data automatically Do not update data automat	Update data automatically O Do not update data automatically		
● Use model time ○ Use cale	endar dates		
First update time:	Q 0 hours ∨		
Update date:	11/22/2023 🔍 🛪 8:00:00 AM 🔹		
Recurrence time:	A 1 hours V		
▼ Data			
Title: Number of coils pro Color: mediumOrchid Value: SlabToCoil.cou Image: Ima	duced v nt()		

Fig.9 Bar chart properties



Fig.10 Bar Chart for total amount of coils produced

Fifth graph: Variations of products at the plant

The given flow chart in the assignment shows the total amount of products produced at the steel plant. However, it is still better to visualize them. Therefore, we added a bar chart and we added 5 products (HR,HRP,CRFH,CRFP,GIGA). Each has a different color to help distinguish between them, the values are different according to the product produced. Notice how at the beginning of the simulation, the amount of products produced is 0 since there is a process that needs to happen before it starts producing the products, but as they move from the conveyor, the products will start to show in the graph and the graph displays the following numbers.

Properties 🛙		1	8 -	
🖢 chart2 - Bar Chart				
Recurrence time:	7	hours	~	^
Data				
Title: HR				
Colon darkMagen	nta 🔍 👻			
Value: outputHR				
Title: HRP				
Color: orange	~			
Value: outputHRP				
Title: CRFH				
Color: darkKhaki	· ·			
Value: outputCRFH	н			
Title: CRFP				
Color: mediumTurqu	uoise. Y			
Value: outputCRFF	Р			
0 2 0 0				v



Fig.11 Bar Chart Properties Fig.12 Different products at the plant

Question 3 (Joelle)

To show the changes in different stages and highlight the different results from various processes, we included two statecharts in this model: one for the ladle and another for the coil.

1. Ladle state chart

To enhance the ladle diagram in our model, we introduced an oval shape at the top. This oval alternates between the colors blue, cyan, dodger blue and yellow Green, representing different phases the ladle undergoes. Our statechart outlines four main states: Delivery, Scrap, Molten Steel, and Post-Degassing, with transitions between these states initiated by specific messages.

The statechart starts in the Delivery state, moving to Scrap upon receiving the "delivering" message. This transition is programmed in the EAF_1 and EAF_2 blocks, where the function `send("delivering", agent)` is activated in the On Enter section. Similarly, the Molten Steel state is initiated by the "melting" message from the LadleFurnace_1 and LadleFurnace_2 blocks.

The final phase, Post-Degassing, is reached from the Molten Steel state through a transition triggered by the "degassing" message. This message is sent by the RH_Plant block using the function `send("degassing", agent)` in its On Enter section.

To visually track these transitions, we implemented the `oval.setFillColor(Color)` function in the Entry and Exit actions of each state. This ensures the oval changes color corresponding to each state transition, vividly illustrating the progress through each step of the process.

	Agent location (delay):
	r Phondes / preemption
Secondary timeMeasureStart CAC 1 LadiaCommon 1	* Advanced
	Customize resource choice:
RH_Plant timeMeasureEnd	Resource selection:
<u>o</u>	Queue: exit on timeout:
ScrapInput1 timeMeasureStart1 EAF_2 LadleFurnace_2	Queue: enable preemption:
	Restore agent location on exit: 📑 🖂
	Force statistics collection: \neg
	* Actions
	On enter: send("delivering", agent)
	On seize unit:
	On enter delay:
	On at exit:
	On exit:
	On remove:

Fig.13 EAF_1 and EAF_2 actions



Fig.14 LadleFurnace_1 action

			Type: ⇒ ⊙ Specified time ○ Until stopDelay() is called Delay time: २ Classifier Classifier
ScrapInput	timeMeasureStart	EAF_1 LadleFurnace_1	Maximum capacity: 🗮 🗹
•	<u>Ğ</u> ⇒ •1*	на по	Agent location: 📑 🔁 InputRH 💙 🛤 😂
			- Advanced
ScrapInput1	timeMeasureStart1	EAF_2 LadleFurnace_2	Forced pushing:
o	@>		Restore agent location on exit: 📃 🗸
			Force statistics collection:
			* Actions
			On enter: send("degassing", agent)
			On at exit:
			On exit:
			On remove:
			* Advanced

Fig.15 RH_Plant action

<pre>< transition -</pre>	Transition	stransition1 -	Transition
Name:	transition Show name Ignore	Name:	transition1 Show name Ignore
Triggered by:	Message v	Triggered by:	Message 🗸
Message type:	Object 🗸	Message type:	Object 🗸
Fire transition:	O Unconditionally	Fire transition:	
	On particular message		On particular message
	O If expression is true		
Message:	"delivering"	Message:	"melting"
message.		Action:	
Action:			
Guard		Guard:	
Guaru.		Description	
Description			
Description			

Fig 16+17 Transition

transition -	Transition	
Name:	transition Show name Ignore	^
Triggered by:	Message V	
Message type:	Object 🗸	
Fire transition:	○ Unconditionally	
	On particular message	
	◯ If expression is true	
Message:	"delivering"	
Action:		
Guard:		
Description		

Name:	transition2 Show name Ignore
Triggered by:	Message V
Message type:	Object 🗸
Fire transition:	O Unconditionally
	 On particular message
	○ If expression is true
Message:	"degassing"
Action:	
Guard:	

Fig 17+18 Transitions

		_
Delivery -	State	
Name:	Delivery Show name Ignore	^
Fill color:	blue 🗸	
Entry action:	oval.setFillColor(blue)	
Exit action:	oval.setFillColor(blue)	
• Description		

Fig.19 Example of State



Fig.20 state chart

2. Coil state chart

Here, we are adding colors to the 3D boxes in our model using a statechart with 6 states and 5 transitions. These transitions are activated by 'String' type messages, set up so that the Fire transition responds to specific messages. The statechart begins at the "Coils" state and includes two transitions activated by messages, leading to either the HR or HRP states. The transition between "Coils" and "HR" is a message and in particular message "HR", As for the transition between "Coils" and "HRP" it is also a message and in particular message "HRP". As for the main chart I put "send("HR", agent)" on enter action in storingDelay. We added "send("HRP", agent)" on enter action in PLTCM. From HRP, transitions lead to either GIGA or CRFH states. For the transition between HRP and GIGA it is a message with one particular message "GIGA" and the same logic for the transition between HRP and CRFH

with the message being "CRFH". As for the main chart I put "send("GIGA", agent)" on enter action in CAL and I added "send("CRFH", agent)" on enter action in BAF. The final transition moves from CRFH to CRFP with a message transition of "CRFP" and "send("CRFP", agent)" as an on enter action in SPM.

The statechart uses colors like powderBlue,mediumPurple,orchid,salmon,lightGrey and yellowGreen. We apply these colors using the `shapeBox.setFillColor(Color)` function in each state's Entry action. This setup ensures that the box colors change dynamically, reflecting the current process stage of the steel and differentiating various outputs by color.

<pre>< transition -</pre>	Transition	
Name:	transition Show name Ignore	^
Triggered by:	Message V	
Message type:	Object 🗸	
Fire transition:	○ Unconditionally	
	On particular message	
	◯ If expression is true	
Message:	"HR"	
Action:		
Guard:		
• Description		

Fig.21 Transition example

SPM - Service Send seized resources:
Agent location (queue):
Agent location (delay):
Priorities / presention
* Advanced
Customize resource choice:
Resource selection:
Queue: exit on timeout:
Queue: enable preemption:
Restore agent location on exit: \neg
Force statistics collection:
* Actions
On enter: send("CRFP", agent)
On seize unit:
On enter delay:
On at exit:
On exit:
On remove:
9 BAF - Delay
Name: BAF Show name Ignore
Type: 🗮 🖲 Specified time
O Until stopDelay() is called
Delay time: d triangular(42,47,55)
Capacity: 3 23
Maximum capacity: 📑 🗌
Agent location: 📑 🔁 BAFCoils 💙 🕏 🛱
* Advanced
Forced pushing:
Restore agent location on exit: 📑 🖂
Force statistics collection:
- Actions
On enter: send("CRFH", agent)
On at exit:
On exit:

P CAL - Service Send seized resources:	
Serve select resources.	311
A	
Agent location (queue):	→ La InputPLTCSM → La
Agent location (delay):	InputCAL V t D
Priorities / preemption	
Advanced	= □
Resource selection:	
Resource selection:	* Nearest to the agent
Queue: exit on timeout:	=, []
Queue: enable preemption:	=, []
Restore agent location on exit:	=, 🗸
Force statistics collection:	=, 🗌
Actions	
On enter: send("GIGA", agent)
On seize unit:	
On enter delay:	
On at exit:	
On exit:	
On remove:	
PLTCM - Service Send seized resources:	2
Agent location (queue):	= 🔁 path7 🗸 🛱
Agent location (delay):	👎 🔁 InputPLTCSM \vee 🐮 🗯
• Priorities / preemption	
* Advanced	
Customize resource choice:	=
Resource selection:	= Off (chooses some matching unit)
Resource selection: Queue: exit on timeout:	=, Off (chooses some matching unit) ▼
Resource selection: Queue: exit on timeout: Queue: enable preemption:	-, Off (chooses some matching unit) -,
Resource selection: Queue: exit on timeout: Queue: enable preemption: Restore agent location on exit:	- Off (chooses some matching unit) -,
Resource selection: Queue: exit on timeout: Queue: enable preemption: Restore agent location on exit: Force statistics collection:	Image: Control of the control of t
Resource selection: Queue: exit on timeout: Queue: enable preemption: Restore agent location on exit: Force statistics collection: Actions	Image: Control of the control of t
Resource selection: Queue: exit on timeout: Queue: enable preemption: Restore agent location on exit: Force statistics collection: Actions On enter: send(<pre> Off (chooses some matching unit) v</pre>
Resource selection: Queue: exit on timeout: Queue: enable preemption: Restore agent location on exit: Force statistics collection: • Actions On enter: Send(On seize unit:	<pre> Off (chooses some matching unit) v</pre>
Resource selection: Queue: exit on timeout: Queue: enable preemption: Restore agent location on exit: Force statistics collection: Actions On enter: Send(On seize unit: Con enter delay: Con enter de	<pre></pre>
Resource selection: Queue: exit on timeout: Queue: enable preemption: Restore agent location on exit: Force statistics collection: • Actions On enter: send(On seize unit: on enter delay: on at exit: i 	<pre> Off (chooses some matching unit) v </pre>
Resource selection: Queue: exit on timeout: Queue: enable preemption: Restore agent location on exit: Force statistics collection: • Actions On enter: send(On seize unit: On enter delay: On at exit: On exit: 	<pre> Off (chooses some matching unit) Off (chooses some matching unit) Off (chooses some matching unit) </pre>
Resource selection: Queue: exit on timeout: Queue: enable preemption: Restore agent location on exit: Force statistics collection: • Actions On enter: send(On enter: send(On enter: send(On enter delay: send(On at exit: send(On exit: send(On exit: send(<pre> Off (chooses some matching unit) v </pre>

() storingDe	lay - Delay
Name:	storingDelay Show name Ignore
Type:	Specified time
	◯ Until stopDelay() is called
Delay time:	✓ triangular(0.5, 1, 1.5) hours ∨
Maximum ca	pacity: 🗦 🔽
Agent locati	on: 号 🔁 receivingNode 🗸 🛱
 Advanced 	
Forced pushi	ng: =, 🗌
Restore agent	: location on exit: 📃 🔽
Force statistic	s collection: =,
• Actions	
On enter:	<pre>send("HR", agent)</pre>
On at exit:	
On exit:	
On remove:	•
 Advanced 	
Agent type:	=↓ 😌 coil 🗸 ∨
 Single age 	nt 🔿 Population of agents
Model/library	Process Modeling Library (change)
Visible:	ves
Visible on	upper agent
Windle Lands	

Figs.22 till Figs 26 are all on enter actions



Fig.27 HRP state example

Fig.28 State charts



Fig.29 Simulation running

Question 4 (Nada)

There are seven essential parameters in our model. They are all represented in the following pictures.

- 1. <u>inputRate -</u> with a default value 2.5 and type double. It controls the rate at which the scrap metal is entering the model and is directly connected to the production rate.
- 2. <u>cranesNumber-</u>represents the total number of cranes in the model, with the value of 10
- 3. <u>ladleWeight-</u> represents the ladle's weight. The default value is 40.
- 4. <u>coilWeight-</u> weight of coil has a default value of 10.
- 5. <u>coilsPerLadle -</u> the amount of coils made per one ladle. The type is double and is connected to ladlesWeight / coilWeight.
- 6. <u>probPLTCM -</u> represents the probability of going to the production of hot rolled steel coils and the rest goes to cold mill. It is also type double and has 0.5 as a default value.
- 7. <u>probCR-by</u> default it is 0.5 probability and determines the specific selectOutput which shows which output is going to be produced.

∂ inputRate	- Parameter		
Name: Visible: Type:	inputRate ves double	✓ S	Show name 🗌 Ignore
Default value:	- 2.5		
System dyr	namics array		
• Value editor			
Label:	inputRate		
Control type:	Text	\checkmark	
Hide condition	ns:		
Parameter	Condition		Value
Advanced			
Static D	ynamic () Actio namics units: pshot	n	
On change:			
• Description			

Fig.30 Input Rate

IadlesWeig	ht - Par	ameter	
Name: Visible:	ladlesW	eight 🗹 :	Show name 🔲 Ignore
Туре:	int	~	
Default value:	- 40		
System dyn	amics arra	у	
 Value editor 			
Label:	ladlesWe	ight	
Control type:	Text	~	
Hide condition	is:		
Parameter	Сог	ndition	Value
 Advanced 			
● Static ○ D	ynamic \subset	Action	
System dyn	amics unit	s:	
Save in sna	pshot		
On change:			
Description			

Fig.31 Ladles Weight

🏽 probCR - P	arameter	
Name: Visible:	probCR ves	Show name Ignore
Туре:	double	~
Default value:	- 0.5	
System dyr	amics array	
• Value editor		
Label:	probCR	
Control type:	Text	\checkmark
Hide condition	IS:	
Parameter	Condition	n Value
 Advanced 		
Static O D	ynamic () Action	'n
System dyr	amics units:	
Save in sna	pshot	
On change:		
Description		

Fig.32 ProbCR

ProbPLTCM	/I - P	arameter			
Name: Visible:	pro	bPLTCM ves		how name 🗌 Ignore	^
Type:	dou	ible	\sim		
Default value:	3	0.5			
System dyn	amics	array			
• Value editor					
Label:	prob	PLTCM			
Control type:	Text		\sim		
Hide condition	s:				
Parameter		Condition		Value	
 Advanced 					
Static D	ynam amics pshot	ic () Action s units:)		
On change:					
 Description 					

const creater i atameter	
Name: coilsPerLadle Show name Ignore	
Visible: Over	
Type: double 🗸	
Default value: 🗧 ladlesWeight/coilWeight	
System dynamics array	
Value editor	
Label: coilsPerLadle	
Control type: Text V	
Hide conditions:	
Parameter Condition Value	
Advanced	
● Static ○ Dynamic ○ Action	
System dynamics units:	
Save in snapshot	
On change:	
Description	

Fig.33 Coils Per Ladle

Fig.34 probPLTCM

	ି coilWeight - Parameter
	Name: coilWeight Show name Ignore
	Type: int 🗸
	Default value: = 10
	System dynamics array
	• Value editor
	Label: coilWeight
	Control type: Text 🗸
	Hide conditions:
	Parameter Condition Value
	Advanced
	Static Upynamic Action
	Save in snapshot
© cranesNumber - Parameter	On change:
Name: cranesNumber Show name Ignore	• Description
Visible: O ves	
Type: int 🗸	
Default value: = 10	
System dynamics array	
* Value editor	Fig.35 Coil Weight
Label: cranesNumber	
Control type: Text 🗸	
Hide conditions:	
Parameter Condition Value	
0 v 4 5 B	
Advanced	
System dynamics units:	
Save in snapshot	
On change:	
* Description	

Fig.36 Cranes Number

Question 5 (Alexandra)

For this task, we will try to make an optimization of our model to increase the production rate by changing the input rate. We create a new model- optimization experiment and set the number of iterations to 500. In the main, we create a new variable 'totalProduction' and in the function body we enter: return (outputHR + 2*(outputHRP + outputCRFH + outputCRFP + outputGIGA))*ladlesWeight: Then, in the optimization we set the objective function to root.totalProduction() and select maximize.

Fig.37 totalProduction Function

Name: Islandedication Value: on Just action (return rothing) Precentation Precentation <th>totalProduc</th> <th>tion - Func</th> <th>tion</th> <th></th> <th></th> <th></th> <th></th>	totalProduc	tion - Func	tion				
Name > ype Ype Faction (vectors nothing) Prective double Arguments Name Ype Faction lody Fraction (outputHR + 2° (outputHRP + outputCRFH + outputCRFP + outputGIGA))*ladleskeight; Advanced Optimization Optimization regine: Genetic Optimization regine: Genetic Objective: Ominimize @ maximize Proct. totalProduction() *umbure of iterations ● Fixed: 500 Infinite Maximum available memory: 512 Main Optimization regine: Optimize Op	Newsy Antolo		Z channe	□			
Note action features nothing) Petures value Type: double action Arguments Name: procention body return (outputHR + 2*(outputCRFP + outputCRFP + outputCIGA))*ladles/eight; Advanced Description Properties ﷺ: Optimization 1 - Optimization Experiment Name: Optimization 1 - Optimization 1 Ignore Top-level agent: Main v Optimization engine: Genetact Objective: minimize @ maximize Prot.totalProduction() Number of iterations © Fixed: 500 Infinite Maximum available memory: \$12 v Mb Create default UI Parameters: Parameters: Parameters: Parameter Type Value Nume: 0.5 BadQualty fixed 10 collweight fixed 10 collweight fixed 10 collweight fixed 10 mumbes fixed 10 Stop time [01/2/2023]** Stop time [01/2/2023]** Stop time [01/2/2023]**	Visible: 00 no	roduction	- Show h	ame 🗀	ignore		
● Returns value Type double ■ Arguments Type ■ Imme Optimization ■ Imme Optimization ■ Imme Optimization ■ Imme Optimization ■ Imme Imme ■ Imme Type ■ Imme Imme ■ Imme Type ■ Imme Type ■ Imme Imme ■ Imme Type ■ Imme Imme ■ Imme Type ■ Imme Type ■ Imme Type	lust action (r	eturns nothing	`				
Type double w Arguments	Returns value	2	,				
Arguments Name Type Function body Peturm (outputHR + 2*(outputHRP + outputCRFP + outputGIGA))*ladleskeight; Advanced Description Properties ﷺ Optimization 1 - Optimization Experiment Name: Optimization 1 - Optimization Experiment Name: Optimization 1 - Optimization Experiment Name: Optimization 0 Optimization engine: Genetic Optimization of iterations ● Fixed: SOO Infinite Maximum available memory: 512 ♥ Mb Create default UI Parameters: Parameters: ImputAte Optimized discrete 0.5 Iadle_light fixed 10 coils_Ladle fixed 10 coils_Ladle fixed 3 MTTF fixed MITH 4 numb_nes fixed 10 Stop if specified time ♥ Stop time: Stop at specified time ♥ Stop time: Stop at specified time ♥ Stop time: Stop date 9/1/2023 ● Stop date 9/1/2023 ●	Type: doubl	e 🗸					
Name Type Image: Standard	Arguments						
I of the set of	Name					Туре	
Production body peturn (outputHR + 2*(outputHRP + outputCRFP + outputGIGA))*ladlesWeight; Advanced Description Properties ﷺ Optimization 1 - Optimization Experiment Name: Optimization 1 - Optimize Imainize Optimization engine: Genetic Objective: Optimize Imainize Imainize Proot.totalProduction() Number of iterations Imainize Prixed: 500 Infinite Maximum available memory: 512 Mainum available memory: 512 Muinum available memory: 512 Muinum available memory: 512 Mainum available memory: 512 Muinum available memory: 512 Mui							
Function body Function body Function body Futurn (outputHR + 2*(outputHRP + outputCRFP + outputGIGA))*ladlesWeight; Advanced Description Properties Properties Properties Properties Properties Prot. totalProduction() Number of iterations Fixed: 500 Infinite Mainwum available memory: 512 ▼ Mb Create default UI Parameters Parameters Parameters Parameters Parameter Type Min Max Step Sued InputRate discrete 10.5 10 0.5 Iadle.jpht fixed 10 coils.Ladle fixed 104 coils.Ladle fixed 104 coils.Ladle fixed 104 Coils.Ladle fixed 10 Coi							
Peturn (outputHR + 2*(outputHRP + outputCRFH + outputGRA))*ladleskeight; Advanced Description Properties % Properties % Optimization1 - Optimization Experiment Name: Optimization engine: Genetic Objective: Ominimize @ maximize root.totalProduction() Number of iterations Infinite Maximum available memory: 512 Main Maximum available memory: 512 Mumber of iterations Fixed: 500 Infinite Parameters Parameters Parameters Parameters Parameters Parameter if ype Min Max Step Sued inputRate discrete 0.5 10 0.5 adleight fixed 10 coilsiadle fixed 3 MTF fixed 1 MTTM f							
Advanced Description Properties 34 Optimization 1 - Optimization Experiment Name: Optimization 1 - Optimization Experiment Name: Optimization engine: Optimization 1 Ignore Top-level agent: Main Optimization engine: Cenetic Too t. totalProduction() Number of iterations Fixed: S0 Infinite Maximum available memory: S12 Mb Create default UI Parameters Parameters Parameters Parameters Parameters Parameter Vue Min Max Step Sued inputRate discrete 0.5 10 0.5 Iadleight fixed 40 coilWeight fixed 40 coilSadle fixed 10 coilWeight fixed 10 coilSadle fixed 3 MTFF fixed 3 MTFF fixed 3 MTFF fixed 1 MTTM	return (ou	tputHR + 2 ³	*(outpu	tHRP +	outputC	RFH + out	<pre>putCRFP + outputGIGA))*ladlesWeight;</pre>
Description Properties St Optimization1 - Optimization Experiment Name: Optimization1 - Optimization I Ignore Top-level agent: Main Objective: Ominimize @ maximize root.totalProduction() Number of iterations Infinite Maximum available memory: 512 Maximum available memory: 512 <td< td=""><td>Advanced</td><td></td><td></td><td></td><td></td><td></td><td></td></td<>	Advanced						
Properties II I Optimization Experiment Name: Optimization 1 Ignore Top-level agent: Main I Optimization engine: Genetic II Objective: Iminimize Imaximize Poot.totalProduction() Number of iterations Image: Imag	Description						
Properties № Optimization1 - Optimization Experiment Name: Optimization1 Ignore Top-level agent: Main Optimization engine: Genetic Optimization engine: Genetic Optimization engine: Genetic Number of iterations ● Fixed: 500 Infinite Maximum available memory: 512 Maximum available memory: 512 Maximum available memory: 512 Marameters Parameters Parameters Parameter Type Min Max Stop: Stop: Stop: Stop: Stop: Stop: Stop: Stop: Stop: Stop date: 10 Stop date: 10/12/2023							
Optimization 1 - Optimization Experiment Name: Optimization engine: Genetic Objective: Indiration engine: Genetic Number of iterations Infinite Maximum available memory: 512 Main Maximum available memory: 512	Properties 8						1 8 1
Name: Optimization 1 Ignore Top-level agent: Main Optimization engine: Optimization engine: Image: Comparison of the state	Optimiza	tion1 - O	ptimiz	ation	Experi	ment	
Main Op-level agent: Main Optimization engine: Genetic Objective: minimize maximize Proct.totalProduction() Number of iterations Infinite Maximum available memory: 512 Main Parameters: Parameters: Parameters: Parameters: Parameter inputRate 0.5 0.6 1.6 1.6 1.6 1.6 1.7	Name:		C	ptimiza	tion1		re
Optimization engine: Genetic Objective: minimize maximize maximize Prost. total Production() Number of iterations Fiked: 500 Infinite Maximum available memory: 512 Maximum available memory: 512 Maximum available memory: 512 Maximum available memory: Fiked: 500 Infinite Main Max Step Sued parameters: Parameters: Parameter Stred Parameter fixed 10 coilWeight fixed 10 coilWeight fixed 10 coilWeight fixed 10 coilWeight fixed 11 fixed 12 Min Max Step Sued inputRate discrete 0.5 probCR fixed 10 start fixed 10 Stop time: 100 Start date: 19/11/2023	Top-level age	ent:	M	lain ∨	1		
Objective:	Optimization	engine:	G	enetic		~	
Corperting Corport root.totalProduction() Number of iterations • Fixed: 500 Infinite Maximum available memory: 512 • Mb Create default UI Parameters Parameters: Value Parameter: Value Parameter: Value Parameter: Value Parameter: Value Parameter: Create default UI Parameter: Value Parameter: Value Parameter: Value Parameter: Value Parameter: Value Pa	Objective:	. enginei		minimi	70 🔘 🛛	avimizo	
Nomber of iterations	Objective.	10			20 0 11	IdAITTIZE	
Parameters: Value Value Parameter Type Min Max Step Sued inputRate discrete 0.5 10 0.5 Interval Interval<	Create defa	ult UI					
Value Value Min Max Step Sued inputRate discrete 0.5 10 0.5 ladleight fixed 40	Parameters						
Parameter Type Min Max Step Sued inputRate discrete 0.5 10 0.5 InputRate InputRate discrete 0.5 InputRate InputRate discrete 0.5 InputRate InputRate discrete 0.5 InputRate discrete discre discrete discrete	Parameters:		Value				
Parameter Type Min Max Step Stued inputKate discrete 0.5 10 0.5 ladleight fixed 40	Description	T	value		Char	Cu. ed	-
Indexight fixed 0.0 0.0 Indexight fixed 10 0.0 cranber fixed 10 0.0 coilWeight fixed 10 0.0 coilWeight fixed 10 0.0 proTCM fixed 0.5 0.0 probCR fixed 0.5 0.0 BadQuality fixed 0.05 0.0 WTTF fixed 1 0 MTTR fixed 1 0 Model time 10 0 0 Stop: Stop at specified time ▼ 100 0 Start time: 0 Stop time: 100 Start date: 19/11/2023 ▼ 5 5	Parameter	Type	IMIN	10	Step	Sued	
Non-Card Fixed 10 coilWeight fixed 10 coilweight fixed 10 coilweight fixed 10 coilweight fixed 10 proTCM fixed 0.5 BadQuality fixed 0.5 BadQuality fixed 0.5 MTFF fixed 3 MTFK fixed 1 MTTM fixed 4 numbnes fixed 10 Stop at specified time ▼ Stop time: 100 Start time: 0 Stop date: 19/11/2023 丁* Nononononononononononononononononononon	ladle ight	fixed	40	10	0.5		
Colliveight fixed 10 collsLadle fixed ladlesWeight/coilWeight proTCM fixed 0.5 probCR fixed 0.5 BadQuality fixed 0.05 reworkTime fixed 3 MTTF fixed 3 MTTF fixed 1 MTTM f	cran ber	fixed	10				
colis.Ladle fixed ladlesWeight/collWeight pro.TCM fixed 0.5 probCR fixed 0.5 BadQuality fixed 0.05 weworKlime fixed 3 MTTF fixed 5 MTTR fixed 4 numbnes fixed 10 Model time Stop: Stop at specified time ▼ Start time: 0 Stop time: 100 Start date: 19/11/2023 ▼ Stop date 19/12/2023 ▼ 000000 ●	coilWeight	fixed	10				
proTCM fixed 0.5 probCR fixed 0.5 BadQuality fixed 0.05 meworkTime fixed 3 MTF fixed 5 MTR fixed 1 MTTM fixed 4 numbnes fixed 10 Model time Stop: Stop at specified time ▼ Start time: 0 Stop time: 100 Start date: 19/11/2023 ▼ Stop date: 19/12/2023 ▼ 0cn0000 ●	coilsLadle	fixed	ladles\	Veiaht/	coilWeig	iht	
probCR fixed 0.5 BadQuality fixed 0.05 reworkTime fixed 3 MTF fixed 3 MTTF fixed 1 MTTM fixed 4 numbnes fixed 10 Model time Stop: Stop at specified time ♥ Start time: 0 Stop time: 100 Start date: 9000000 Stop date: 19/12/2023	proTCM	fixed	0.5			,	
BadQuality fixed 0.05 reworkTime fixed 3 MTTF fixed 3 MTTR fixed 1 MTTR fixed 1 MTTM fixed 4 numbnes fixed 10 Model time Stop tspecified time ▼ Stop: Stop at specified time ▼ Start time: 0 Stop time: 100 Start date: 19/11/2023 丁 Stop date: 19/12/2023	probCR	fixed	0.5				
reworkTime fixed 3 MTTF fixed 5 MTTR fixed 1 MTTM fixed 4 numbnes fixed 10 Model time Stop at specified time ▼ Start time: 0 Start time: 0 Start time: 0 Stop date: 19/12/2023 Topopono 0	BadQuality	fixed	0.05				
MTTF fixed 5 MTTR fixed 1 MTTM fixed 1 MTM fixed 4 numbnes fixed 10 Model time	reworkTime	fixed	3				
MTR fixed 1 MTIM fixed 4 numbnes fixed 10 Model time Image: Stop at specified time ▼ Image: Stop at specified time ▼ Start time: 0 Stop time: 100 Start time: 0 Stop date: 19/12/2023 ▼ 0 0000000 0000000 Image: Stop date: 19/12/2023 ▼	MTTF	fixed	5				
MTTM fixed 4 numbnes fixed 10 Model time Stop: Stop at specified time ▼ Start time: 0 Stop time: 10 Stop date: 19/12/2023	MTTR	fixed	1				
numbnes fixed 10 Model time Stop: Stop at specified time ♥ Start time: 0 Stop time: 100 Start date: 19/11/2023 ■♥ Stop date: 19/12/2023 ■♥	MTTM	fixed	4				
Model time Stop: Stop at specified time ▼ Start time: 0 Stop time: 100 Start date: 19/11/2023 ■▼ Stop date: 19/12/2023 ■▼	numbnes	fixed	10				
Stop: Stop at specified time ▼ Start time: 0 Stop time: 100 Start date: 19/11/2023 0000000 0000000	Model time						
Start time: 0 Stop time: 100 Start date: 19/11/2023 * Stop date: 19/12/2023	Stop:	Stop at spec	ified tim	e v			
Start date: 19/11/2023 Stop date: 19/12/2023 COORD	Start time:	0				Stop time	: 100
	Start date: 1	9/11/2023				Stop date	19/12/2023
	6	0:00:00					00:00:00 +

Fig.38 Optimization Parameters

Running the model, we observe that with the best input rate equal to 7, we get a maximum total production of steel equal to 56,480 with 17 iterations.

SteelManufacturingPlant1 : Optimization1



Fig.39 Optimization Result Model

Question 6 (Nada+Alexandra)

We need to simulate a breakdown and maintenance system of only 2 cranes. A breakdown of the crane may happen every 5 days on average and there is maintenance happening regularly every 4 days. We added three states, InOperation, maintenance and failure. We also added the parameters MTTR, MTTM, MTTF and numberofcranes as shown in the figures below. From InOperation to maintenance a transition was added with a rate of 1/MTTM per day and an action of Cranes.set_capacity(2). From maintenance to InOperation it was a rate of 1/MTTR per day. From inOperation to failure a transition with a rate of 1/MTTF per day was added with an action of Cranes.set_capacity(2) and lastly the transition from failure to inoperation it was a rate of 1/MTTR per day. As for the states we put the capacity of In Operation as the total number of cranes by setting an action of Cranes.set_capacity(numberofcranes).

InOperation Failure	echart1	 MTTR MTTM MTTF numberofcranes 		
InOperation	on - State			
Name:	InOperation Sho	ow name 🗌 Ignore		^
Fill color:	yellowGr 🗸			
Entry action:	Cranes.set_capacit	cy(numberofcranes)]
Exit action:]
• Description]
Description				
■ Properties 🛛			2	
Failure - S	itate			
Name:	Failure Sh	ow name 🗌 Ignore		^
Fill color:	red 🗸			
Entry action:]
Exit action:]
• Description				
Properties 🛛			Č Š	
Maintena	nce - State			
Name:	Maintenance 🗹 Sho	ow name 🗌 Ignore		^
Fill color:	gold 🗸			
Entry action:				
Exit action:				
• Description	-			

Fig 40 to Fig 43 State chart representing the maintenance system and each of the states with their specific colors

	MTTF - Parameter
© MTTR - Parameter	Name: MTTE Show name Ignore
Name: MTTR Show name Ignore	Visible: 💿 ves
Visible: 💿 ves	Type: double ¥
Type: double 🗸	
Default value: 🗧 1	Default value:
System dynamics array	System dynamics array
 Value editor 	Value editor
Labah MTTD	Label: MTTF
	Control type: Text
Control type: lext	Hide conditions:
Hide conditions:	Parameter Condition Value
Parameter Condition value	
× Advanced	Advanced
	Static Dynamic Action
	System dynamics units:
Save in snapshot	Save in snapshot
On change:	On change:
* Description	Description
	•
C MTTM - Parameter Name: MTTM Show name Ignore C m Visible: © vrs N N N Type: double ✓ N N Default value: = 4 17 Ty System dynamics array > Value aditor 0	umberofcranes - Parameter ame: numberofcranes S Show name Ignore sible: ves upe: int v fault value: 1 10 System dynamics array
	halt
Control type: lext	
Parameter Condition Value	ide conditions:
	arameter Condition Value
* Advanced	
Static O Dynamic O Action	dvanced
System dynamics units:) Static 🔿 Dynamic 🔿 Action
Save in snapshot	System dynamics units:
On change:	Save in snapshot
• Description	n change:
- D	escription

Fig 44 to Fig 47 Parameters

<pre>< transition3</pre>	- Transition	
Name:	transition3 Show name Ignore	
Triggered by:	Rate 🗸	
Rate:	☑ 1/MTTM per day ∨	
Action:	Cranes.set_capacity(2)	
Guard:		
Description		

Fig 48 (Transition 3)-From InOperation to Maintenance:

Fig 49 (Transition 4)- From Maintenance to InOperation

<pre>< transition4</pre>	- Transition	
Name:	transition4 Show name Ignore	
Triggered by:	Rate 🗸	
Rate:	⊋ 1/MTTR per day ¥	
Action:		
Guard:		
• Description		

Fig 50 (Transition 2) from InOperation to Failure:

Properties 🛛		8		
transition2 - Transition				
Name:	transition2 Show name Ignore		^	
Triggered by:	Rate V			
Rate:	☑ 1/MTTF	per day 🗡		
Action:	Cranes.set_capacity(numberofcranes - 2)			
Guard:				
• Description				

Fig 51 (Transition 5) From Failure to InOperation

transition5 - Transition				
Name:	transition5 Show name Ignore	^		
Triggered by:	Rate 🗸			
Rate:	↓ I/MTTR per day ♥			
Action:				
Guard:				
Description				

Question 7 (Hala)

For this question, we need two flows, one for the quality check and the other for reworking. The total amount of slabs are entered into the flow, we use a variable called "Total Slabs" and give it a function "LadleToSlab.count()", because we want it to count all slabs produced in the steel plant. Only 5% of the total amount of slabs are counted as bad, therefore we set a parameter called "BadQuality" and we set it to be double, and 0.05. Quality check is the inflow of "total slabs* bad quality".



Fig 52 System dynamics model

O TotalSlabs - Dynamic Variable	
Name: TotalSlabs	
Show name 🗌 Ignore 🗌 Visible on upper	agent
Visible: 💿 yes	
Color: Default 🗸	
Array Dependent Constant	
TotalSlabs=	
LadleToSlab.count()	

Fig 53 Varibale counting slabs

	© BadQuality - Parameter	
	Name: BadQuality ✓ Show name Ignore Visible: ● ves Type: double ✓ Default value: ● . 05 System dynamics array ✓ Value editor	
Value editor		

Fig 54 - 55 Fixed Parameters for rework and bad quality

The stock for badSlabs is 0 at the beginning since no bad slabs are entered but as the simulation starts, the stock will start to get filled. The bad slabs are checked and then will go to rework, rework takes 3 hours, so the parameter is set to 3 and our model time units are in hours. Reworking is the inflow of only bad slabs and the time it takes 3 hours to fix them, hence reworking is calculated as "bad slabs/ reworkTime".

All parameters and the variable were joined by links to the flow.



Fig 56 Bonus question (Hala+Joelle):

To extend our model with more parameters and feedback loops, we decided to turn slabs to steel plates. We will only need good slabs for that, therefore we will take 90% of the good slabs produced and we will take 35% of the number of refurbished slabs produced and put them in our stock. Only 50% of the stock will be used to produce the plates.

We add a flow connecting before quality check, it will take the total amount of slabs produced and then moves through a flow that takes the 90% as good slabs produced, keeps them in inventory which is the stock here, and then we added another flow after refurbished slabs, the flow will take only 35% of refurbished slabs, the parameter of "RefurbishedToStock" is set to 0.35. The amount is also added to the stock.

Finally we added a final flow from the stock to produce the steel plates, we are not using our whole stock to produce the steel plates but rather only 50% of it, hence we set the parameter "GoodToPlates" to be 0.50.



Fig. 57 Extended Model



Fig. 58 Simulation of Extended Model

The simulation shows that only 90% of total slabs are considered to be good and will be moved to the stock, the amount of refurbished slabs that have finished with rework and will go to the stock is only 35%. Finally, the model shows that only 50% of the stock will be made into steel plates.

All necessary parameters are equations for the flow of the extended model are provided below

C GoodQuality - Parameter	⇔ flow - Flow
Name: GoodQuality Ignore Visible: ♥ yes Type: double Default value: ● 0.90 System dynamics array	Name: flow Show name Ignore Visible on upper agent Visible: • yes Color: Default • Array Dependent Constant flow= GoodSlabs*GoodQuality
C RefurbishedToSlabs - Parameter	Same: flow1
Name: RefurbishedToSlabs Show name □ Ignore Visible: • yes Type: double • Default value: = 0.35 □ System dynamics array	Show name ☐ Ignore ☐ Visible on upper agent Visible:
Image: GoodToPlates - Parameter Name: GoodToPlates	Flow2 - Flow Name: flow2 Show name Ignore Visible: yes Color: Default
□ Ignore Visible: ● yes Type: double ▼ Default value: = 0.50 □ System dynamics array	☐ Array ☐ Dependent ☐ Constant flow2= GoodToPlates*stock

Fig 59-64 parameters and flows added to the extended model

Conclusion (Nada)

In this assignment, we combined all of the previously learned skills in AnyLogic software, from building the model to making state charts. It represents the whole process of steel manufacturing and simulates the organizational process of a steel factory, with all of its main parameters.