# **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 23		8	Ш
🖾 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			
▼ Data update			

Fig.3 Time plot properties



Fig.4 Amount produced in tons over time

🔲 Properties 🛛		000	- 0
🌆 chart - His	togram		
Name: cha Show PDF	rt ☐ Ignore ☑ Visible on upper ag	ent	□
Title: Histogram:	Slab processing time		
PDF color:	green CDF color:	1	M
CDF & mean	Iline Width:         V         1           Image: Second Seco		_
🗣 Add histog	ram data		
<ul> <li>Data update</li> <li>Appearance</li> </ul>			



#### **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	ically
● 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 🛙		<b>1</b>	-	
🖢 chart2 - Bar Chart				
Recurrence time:		hours	~	^
+ Data				
Title: HR				
Colon darkMagen	ta 🗸 🗸			
Value: outputHR				
Title: HRP Color: orange	<b>~</b>			
Value: outputHRP				
Title: CRFH				
Color: darkKhaki	¥			
Value: outputCRFF	1			
Title: CRFP				
Color: mediumTurqu	ioise] Y			
Value: outputCRFF	,			
0 2 0 0				~
			>	



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 LadiaGuerran 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 Telay time: Triangular(4*ladlesWeight, 4,
ScrapInput	timeMeasureStart	EAF_1 LadleFurnace_1		Maximum capacity: 🔍 🖂
••	<u> </u>		RH Plant timeMeasureEnd	Agent location: 🗁 🔁 InputRH 💙 ঝ 🖽
		$\sim$	<u>→ġ</u>	* Advanced
ScrapInput1	timeMeasureStart1	EAF_2 LadleFurnace_2		Forced pushing:
<u>.</u>	<u>\$</u> *			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>&lt; 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
	<ul> <li>On particular message</li> </ul>
	○ 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>&lt; 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):	T InputCAL V 15 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: → □	
Agent location (queue):	= 🔁 path7 🗸 🛱
Agent location (delay):	👎 🔁 InputPLTCSM \vee 🏗 🗯
Priorities / preemption	
* Advanced	
Customize resource choice:	5 □
Resource selection:	□ = Off (chooses some matching unit) ∨
Resource selection: Queue: exit on timeout:	□ □ Off (chooses some matching unit) v =, □
Resource selection: Queue: exit on timeout: Queue: enable preemption:	Image: Control of the second secon
Resource selection: Queue: exit on timeout: Queue: enable preemption: Restore agent location on exit:	Image: Contract of the system       Image: Contract of the system <td< td=""></td<>
Resource selection: Queue: exit on timeout: Queue: enable preemption: Restore agent location on exit: Force statistics collection:	-     Off (chooses some matching unit)       -     -       -     -       -     -       -     -       -     -       -     -       -     -       -     -       -     -       -     -       -     -
Resource selection: Queue: exit on timeout: Queue: enable preemption: Restore agent location on exit: Force statistics collection: Actions	Image: symplectic condition     Image: symplectic condition       Image: symplectic conditic conditic conditicon     Image: symplectic conditicon<
Resource selection: Queue: exit on timeout: Queue: enable preemption: Restore agent location on exit: Force statistics collection: Actions On enter: send(	Image: symbol of the symbol
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>F Off (chooses some matching unit) V F. F.</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:	<pre>     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: <ul> <li>send(</li> <li>On seize unit:</li> <li>on enter delay:</li> <li>on at exit:</li> <li>interval</li> </ul>	<pre></pre>
Resource selection:         Queue: exit on timeout:         Queue: enable preemption:         Restore agent location on exit:         Force statistics collection:         • Actions         On enter: <ul> <li>send(</li> <li>On exit:</li> <li>On at exit:</li> <li>On exit:</li> <li>Image: Collection:</li> </ul>	<pre></pre>
Resource selection:         Queue: exit on timeout:         Queue: enable preemption:         Restore agent location on exit:         Force statistics collection:         • Actions         On enter: <ul> <li>send(</li> <li>on exit:</li> <li>on at exit:</li> <li>on exit:</li> </ul>	<pre></pre>

	۱
<sup>©</sup> 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	apacity: 🗦 🔽
Agent locati	on: 📑 🔁 receivingNode 💙 🎀 😂
<ul> <li>Advanced</li> </ul>	
Forced pushi	ng: =, 🗌
Restore agent	t location on exit: 📃 🔽
Force statistic	es collection: 🔍 🗌
Actions	
On enter:	<pre>send("HR", agent)</pre>
On at exit:	
On exit:	
On remove:	
<ul> <li>Advanced</li> </ul>	
Agent type:	≂ <mark>e</mark> coil ∽
<ul> <li>Single age</li> </ul>	ent 🔿 Population of agents
Model/library	Process Modeling Library (change)
Visible:	ves
Visible on	upper agent
You have a second	

## Figs.22 till Figs 26 are all on enter actions



Fig.27 HRP state example

Fig.28 State charts