

LIMBO Tutorial An Eclipse-based Tool for Modeling of Load Variations

Jóakim Gunnarsson v. Kistowski

Department of Computer Science Chair for Computer Science II Software Engineering

July 17, 2017 LIMBO version 0.16.1.201606230914

Contents

1	LIM	BO Tutorial	1
	1.1	Installing LIMBO	1
	1.2	Installation via Update Site	1
		1.2.1 Building LIMBO from Code	2
	1.3	Creating a new Model	2
		1.3.1 Modifying the Seasonal Part	4
		1.3.2 Modifying the Trend Part	5
		1.3.3 Modifying the Burst and Noise Parts	6
	1.4	DLIM Editor	7
		1.4.1 Plot View	8
		1.4.2 Editing a DLIM instance in the Editor	8
		1.4.3 Generating Time Stamps 1	1
		1.4.4 Extracting a DLIM Sequence from a Trace	1
		1.4.5 Comparing Model and Trace	2
	1.5	Additional Features	3
		1.5.1 Periodic Process Extractor	3
		1.5.2 Difference Calculator	4
	1.6	Example Models	5

1. LIMBO Tutorial

LIMBO requires an up-to-date version of the Eclipse Modeling Tools. It has been tested with Eclipse Kepler, Luna, Neon, and Oxygen. LIMBO should also work with any newer Eclipse Version. We generally recommend using the newest available version. The newest version at the time of the latest update to the tutorial is available at:

http://www.eclipse.org/downloads/packages/eclipse-modeling-tools/oxygenr

Eclipse must be running using Java 6 or newer.

1.1 Installing LIMBO

There are two ways to gain access to LIMBO:

1.2 Installation via Update Site

LIMBO can be downloaded from its Eclipse Update Site at:

http://se2.informatik.uni-wuerzburg.de/eclipse/limbo/

To use this site click on $\text{Help} \rightarrow \text{Install new Software ...}$ in the Eclipse IDE, then click the Add... Button and enter the URL there. The site can then be selected in the Work with: drop-down menu and the feature should appear (Fig. 1.1).

0		Inst	all		– 🗆 🗙
Available S Check the i	oftware tems that you wish to instal	I.			
<u>W</u> ork with:	LIMBO - http://se2.inform			bo/ 🗸	<u>A</u> dd <u>Sites"</u> preferences.
type filter te	xt				
Name	IMBO			Version	ı
	B LIMBO			0.14.12	.1013
< <u>S</u> elect Al	Deselect All	1 item selected			>
Details LIMBO: Loa	d Intensity Modeling Tool				1
Show only	the <u>l</u> atest versions of availa	able software	✓ <u>H</u> ide items that	it are already instal	led
✓ <u>G</u> roup iter	ns by category		What is <u>already</u>	installed?	
Show only	software applicable to targ	et environment			
✓ Contact a	l update sites during install	to find required	software		
?		< <u>B</u> ack	<u>N</u> ext >	<u>F</u> inish	Cancel

Figure 1.1: The Install new Software ... dialog, with LIMBO selected.

1.2.1 Building LIMBO from Code

The feature can also be built directly from Code, available on GitHub at

https://github.com/joakimkistowski/LIMBO

by checking out the following plugin projects:

```
https://github.com/joakimkistowski/LIMBO/tree/master/dlim.exporter
https://github.com/joakimkistowski/LIMBO/tree/master/dlim.extractor
https://github.com/joakimkistowski/LIMBO/tree/master/dlim.generator.edit
https://github.com/joakimkistowski/LIMBO/tree/master/dlim.generator.editor
https://github.com/joakimkistowski/LIMBO/tree/master/dlim.generator
```

LIMBO can then be executed by right-clicking on any of the projects and choosing Run $As \rightarrow Eclipse Application$.

Please note: This tutorial is a LIMBO user tutorial. If you intend to develop Eclipse plugins for LIMBO or using LIMBO, please refer to the LIMBO Architecture description here:

http://se2.informatik.uni-wuerzburg.de/pa//uploads/papers/paper-771.pdf. An even further in-depth description of LIMBO's code structure can be found in here: http://se2.informatik.uni-wuerzburg.de/pa//uploads/papers/paper-749.pdf (pages 37-51)

1.3 Creating a new Model

A new Descartes Load Intensity Model can be created within the context of any Eclipse Project of any project type (Using separate projects for DLIM modeling is recommended). To create a new model instance click $File \rightarrow New \rightarrow Other$; in the dialog choose: Descartes Load Intensity Model \rightarrow Descartes Load Intensity Model and click Next > (Fig. 1.2). Now select the project in which to place the model and enter a name.

The model creation wizard (first page in Fig. 1.3) allows for easy creation of an initial model with the use of parameters (as defined by the hl-DLIM meta-model) for the different parts of the model. This will be done during the course of this tutorial. Click on Next > to get to the next wizard page. It can however be skipped at any point by clicking the **Finish** button.

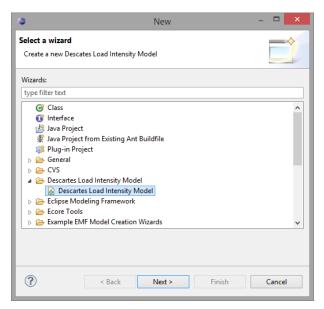


Figure 1.2: Choosing the DLIM creation wizard.

Nev	· – –	×
Descartes Load Intensity Model		
Create a new Descates Load Intensity Model		
Enter or select the parent folder:		
LIMBO_Tutorial_Project		
😂 LIMBO_Tutorial_Project		
File name: exampleModel.dlim		
Advanced >>		
? < Back Next	> Finish Cance	el

Figure 1.3: Creating a new model.

The next dialog page (Fig. 1.4) offers a choice about which model parts to edit in the

wizard. We can extract the wizard's parameters from an arrival rate trace and modify the seasonal, trend, burst, and noise parts of the model. For now we leave this page at its default settings (Extract Model Parameters from Trace unchecked, everything else: checked) and click on Next >.

a	New	- 🗆 🗙
	el Parameters Initial Model Parameters to be set.	
Modify Se		
?	< Back Next > Finish	Cancel

Figure 1.4: Choosing which parameters to edit.

1.3.1 Modifying the Seasonal Part

This dialog page (Fig. 1.5) offers to define the model's seasonal part. The seasonal part is the repeating base function of the model. It is defined by its arrival rate peaks and base values, as well as its period (duration of a single seasonal iteration). I recommend playing around with the parameters to get a feel for them. In the end set them as follows:

Period:	24
Number of Peaks:	2
Base Arrival Rate Level:	2
Base Arrival Rate Level between Peaks:	4
First Peak Arrival Rate:	12
Last Peak Arrival Rate:	11
Interval containing Peaks:	12
Seasonal Shape:	SinTrend

Then click on Next >.

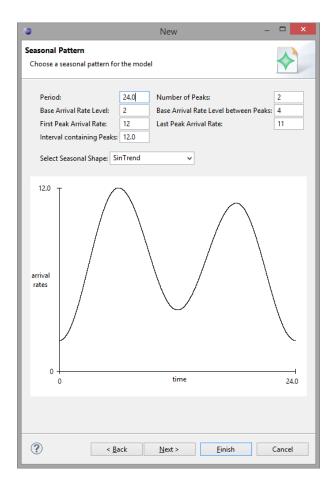


Figure 1.5: The Seasonal Part.

1.3.2 Modifying the Trend Part

This dialog page (Fig. 1.6) defines the model's trend part. The trend part defines a piecewise function, which interpolates at the maximal seasonal peaks so that these peaks reach the target arrival rate defined in the list view. Each trend segment stretches over multiple seasonal iterations and interpolates between these defined target peaks.

The trend segment length is defined by the Number of Seasonal Periods within one Trend. Set this to 2.

Next we must define the target arrival rates which the seasonal peaks are to reach:

In the text-field next to Interpolate max. seasonal peak to target arrival rate: enter 12, then klick the Add button. The first trend segment now begins at the biggest peak of the first seasonal iteration. This peak has an arrival rate of 12.

Next enter 20 in the same text-field and click Add again. The first trend segment will end by interpolating the maximum arrival rate of its last seasonal iteration (remember: the segment stretches over 2 seasonal iterations) to the arrival rate of 20.

At last enter 16 and click Add again.

As the **Trend Shape** select **SinTrend**. This is the function the trend uses for interpolation between its defined arrival rates.

🗢 New – 🗆 🗙
Overarching Trends
Choose the overarching trends for the model
Number of Seasonal Periods within one Trend: 2 Interpolate max. seasonal peak to target arrival rate using Trend: 12.0 20.0 16.0
Interpolate max. seasonal peak to target arrival rate: 16 Add Select Trend Shape: SinTrend v Select Operator: MULT v Explicitly show Trend Contribution in Plot
arrival rates 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Figure 1.6: The Trend Part.

Click Next >.

1.3.3 Modifying the Burst and Noise Parts

The last dialog page (Fig. 1.7) offers the definition of recurring bursts and random noise. Both are added onto the existing arrival rate output.

Define the bursts as follows:

First Burst Offset:18Inter Burst Period:72Burst Peak Arrival Rate:10Burst Width:4

Additionally set the Maximum Noise Arrival Rate to 3.

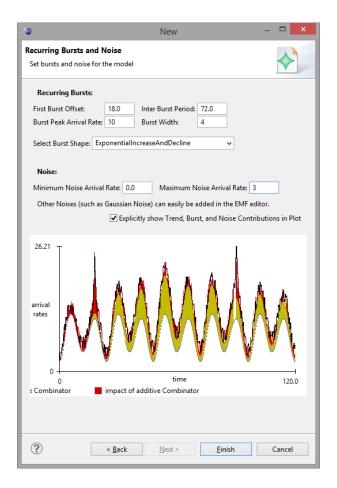


Figure 1.7: The Burst and Noise Parts.

You are now done. Click **Finish** to exit the wizard.

1.4 DLIM Editor

The DLIM Editor opens automatically (Fig. 1.8). The model should already be prepopulated with a root *Sequence*, a number of *TimeDependentFunctionContainers*, and a few *Combinators*.

It is recommended to turn on Live Validation for easy modeling feedback, by right-clicking inside the editor and checking **Live Validation**. Model element attributes can be changed in the Properties View, which can be opened by right-clicking on any model element (such as the root *Sequence*) and selecting **Show Properties View**. You should also open the Plot View, which visualizes the model's current arrival rate function. Do this by right-clicking anywhere in the editor and the clicking on **Show Plot View**.

Rearrange the Plot View and Properties View so that both are accessible at the same time.

•	Resource - LIMBO_Tutorial_Project	ct/exampleModel.dlim - Eclipse 📃 🗖	I X
File Edit Navigate Search Project	Dlim Editor Run Window Help		
11 • 8 6 4 9 9 • 6 4	$\bullet \underline{b} \bullet \overline{b} \bullet \phi \bullet \bullet \bullet \underline{a} = 0$	Quick Access 📄 🛱 Java 💦 Resource 🛗 SVN Repository Exp	ploring
Project Explorer 🛛 👘 🗖	log exampleModel.dlim ⊠		
🖻 🐄 👘 🍸	Resource Set		
▲ LIMBO_Tutorial_Project	a platform:/resource/LIMBO_Tutorial_Press	oject/exampleModel.dlim	
axampleModel.dlim	Sequence exampleModel		
	 Combinator MULT Combinator ADD 		
	Combinator ADD		
	Time Dependent Function Cont	ainer seasonal0	
	Fime Dependent Function Cont	ainer seasonal1	
	b Image Dependent Function Cont		
	Time Dependent Function Cont	ainer seasonal4	
	Selection Parent List Tree Table Tree wit	h Columns	
	Properties 🛛	· · · · · · · · · · · · · · · · · · ·	
	Property	Value	
	Name	E exampleModel	
	Terminate After Loops	1 5	
	Terminate After Time (s)	⁸⁴⁴ 0.0	
< >>			
< >			
🕗 Tasks 🔵 Arrival Rate Plot 🕴		·	
26.19 -			
A			
arrival	المعتمر المعتمل المعلم	all and the second an	
rates	North Margaret L. M. "	have the of the out the over the work in	۲ <u>۱</u>
ya - v	yy hugo	and the second sec	'Vin
0		time	120.0
exampleModel			12010
Selected Object: Sequence exampleModel			

Figure 1.8: The initial model.

1.4.1 Plot View

Right-clicking in the Plot View offers a few options. You can save the current plot to a file or display arrival rates from a trace for comparison.

For now, toggle the plot decomposition by clicking on **Toggle Decomposition**. The decomposition shows the impact of the different *Combinators* on the total arrival rate function.

The plot view context menu also offers options to save the displayed plot as a .png image, and it can optionally display arrival rates as defined by an arrival rate trace (see Section 1.4.5).

1.4.2 Editing a DLIM instance in the Editor

All functions displayed in the DLIM editor can be deleted or edited. For this tutorial we are going to delete the uniform noise function and replace it with a normal noise distribution. We are then going to multiply a linear function onto this noise, so that it is strongest at the beginning and then fades out towards the end.

The Uniform Noise is contained in the third Combinator (The second Combinator ADD). Open this Combinator, then click on the Uniform Noise and delete it. The editor will now display an error, if Live Validation is enabled (Fig. 1.9).

눱 Resource Set							
a 🔬 platform:/resource/LIMBO_Tu	utorial_Project/exampleModel.dlim						
a 😰 Sequence exampleModel							
Combinator MULT							
b							
🕪 Combinator ADD							
b Ime Dependent Function							
b Image Dependent Function							
b Ime Dependent Function							
Fine Dependent Function	tion Container seasonal4						
election Parent List Tree Table	Tree with Columns						
	1						
Properties 🖂		🛃 🖬 🎲 💷 🕼 🔍 🗖					
roperty	Value						
Operator	I≡ ADD						
•							
•							
•							
•							
•							
•							
•							
${\frown}$							
ion impact of multiplicati		ditive Combinator					

Figure 1.9: Our model with the Uniform Noise deleted.

Next add a Normal Noise to the Combinator. For this right-click on the Combinator's then New Child \rightarrow Normal Noise (Fig. 1.10).

9	Re	source - LIMBO_Tutorial_Project/exampleModel.dlim - Eclipse	- 🗆 🗙
File Edit Navigate Search Project Dlin	m Editor Rur	Window Help	
i 📬 🕶 🗟 🖉 💁 🖉 🗸 🕶	철 - 원 -	··· ← ← • → • Quick Access	a 🔁 Resource 🔠 SVN Repository Exploring
Project Explorer 🛛 " 🗆	Resource	eNodel.dlim 🙁 e Set form:/resource/LIMBO. Tutorial. Project/exampleModel.dlim	
		equence exampleModel	
	⊳		juence
	Þ		iform Noise
	Þ	Undo Delete Ctrl+Z	rmal Noise
	×	Kedo Ctri+Y Y	nstant
		Copy 🛠 Exp	ionential Increase And Decline ionential Increase Logarithmic Decline ear Increase And Decline
		🗶 Delete 🕸 Ab:	solute Sin
	Selection	Validate Live Validation Control	ear Trend oonential Trend garithmic Trend
	Property Operat	Debug As Arr Run As Arr Team Compare With	Trend Ival Rates From File solute Value Function ynomial
		Replace With	
< >		Load Resource	
🖉 Tasks 🧶 Arrival Rate Plot 🕴		Refresh Show Properties View	
arrival rates		Remove from Context Ctrl+Alt+Shift+Down Show Plot View	\mathbf{M}
0		time	120.0
exampleModel	····· original	function impact of multiplicative Combinator impact of additive	Combinator
Selected Object: Combinator ADD			

Figure 1.10: Creating a new model element.

To edit the new *Normal Noise* select it (click on it), then change its attributes in the Properties View. Set its **Mean** to **5** and its **Standard Deviation** to **3** (Fig. 1.11).

📓 *exampleModel.dlim 🖾											Ē				
🍋 Resource Set															
	Sequer Co Co Co Co Co Co Co Co Co Co	nce exa mbinat mbinat mbinat Norma ne Dep	mpleM or MUL or ADD or ADD al Noise endent	odel .T : 5.0 Funct	torial_Project/e tion Container si	asonal	D	lim							
D Selection	🔶 🔶 Tin	ne Dep	endent	Funct	tion Container se tion Container se Tree with Colu	asonal									
Prope	rties 🖂								1	Ŀ	*	I (1	ç, 🗸	·	1
Property						Value				_					
Mean Standard Deviation					Las 5.0										
					LT: 3.0										
											_				
<															

Figure 1.11: Editing the Normal Noise.

Next we add a *Combinator* to the *Normal Noise*. Right-click on the *Normal Noise* \rightarrow **New Child** \rightarrow **Combinator**. Set the new *Combinator*'s **Operator** to **MULT** in the Properties View.

We now add a *Linear Trend* to the new *Combinator*. Right-click on the *Combinator* \rightarrow **New Child** \rightarrow **Linear Trend**. In the Properties View set the *Linear Trend*'s **Function Output At Start** to **1** and its **Function Output At End** to θ .

We have now successfully replaced the original *Uniform Noise* with a linearly diminishing *Normal Noise* (Fig. 1.12).

\$	Resource - LIMBO_Tutorial_Project/exam	pleModel.dlim - Eclipse 🛛 🗕 🗖 🗙								
File Edit Navigate Search Pro	oject Dlim Editor Run Window Help									
📑 🕆 🗄 🕼 📥 🔊 💁 🕶 🖉	∋ 🖋 • ½ • № • ♥ • • • • ≅ -									
	Quick Access	😰 🖹 🐉 Java 👔 Resource 💀 SVN Repository Exploring								
Project Expl ☆ □ □ □ ☆ □ ↓ ↓ ₩ LIMBO_Tutorial_Project ↓ ₩ wampleModel.dlim		cct/exampleModel.dlim								
R example woder.dim	 Sequence exampleModel Combinator MULT Combinator ADD Combinator ADD 									
	 Normal Noise 5.0 Combinator MULT Linear Trend 1.0 									
	 Time Dependent Function Contain Time Dependent Function Contain 									
	A Time Dependent Function Contain A Time Dependent Function Contain									
	b Ime Dependent Function Contain	ner seasonal4								
	Selection Parent List Tree Table Tree with	Columns								
	■ Properties ⊠	🛃 🖬 🎠 🖾 🕨 🗖 🗖								
	Property	Value								
	Function Output At End Function Output At Start	E13 0.0								
	Tunction output At start									
< >	<	>								
🚈 Tasks 🌘 Arrival Rate Plot 🔀										
26.74 -	والمتريب والقلاب والملك والملكية ومناكر والمكتم والمناج والترجي والمراجع									
rates										
0 exampleiのgiginal function	time impact of multiplicative Combinator	mpact of additive Combinator 120.0								
		1								
L										

Figure 1.12: The edited Model.

1.4.3 Generating Time Stamps

Once no validation errors appear and the model has been saved (**ctrl+s**), a request timestamp series can be generated by right-clicking the .dlim model file in the Eclipse Package Explorer and selecting **Generate Time Stamps** (Fig. 1.13). A list of all currently installed time-stamp exporters appears. All default exporters, shipped with LIMBO write their resulting time series to their respective folders within the model's Eclipse project.

	Resource - LIMBO	Futorial_Project/exampleMod	del.dlim - Eclipse 🛛 🗕 🔍
File Edit Navigate	e Search Project Dlim Editor Run		
1 - H R A	🖉 💁 🖌 🧀 🛷 = 👌 = 🖗 =	5 6 + 5 + e	
		Quick Access	🐉 Java 🔥 Resource 🔠 SVN Repository Exploring
Project Expl 2		n 🛛 nurce/LIMBO_Tutorial_Project/exam exampleModel F3 F3	pleModel.dlim
	Image: Copy Image: Copy <	Ctrl+C Ctrl+V Delete Ctrl+Alt+Shift+Down Ctrl+Alt+Shift+Up	а0 а1 а2 а4
	Export		M ~ = 0
	Refresh Debug As Run As Team Compare With Replace With	F5	
<	Properties	Alt+Enter	
🖉 Tasks 🥥 Arriva		e File	
26.74 arrival rates	Generate Time Stamps	time	The second secon
example PRiginal f			additive Combinator
exampleModel.dli	im - LIMBO_Tutorial_Project		

Figure 1.13: Generate Time-Stamps.

For this tutorial we want to create request time stamps for the use with a benchmarking framework. For this, the time-stamp generator samples the arrival rate function and then generates time-stamps according to the sampled arrival rate within each sampled interval. Select **Request Time Stamps via Equal Distance Sampling**, then click **OK**. This creates the request time-stamps with an equal distance from each other within each sampled arrival rate interval.

The resulting dialog offers a number of parameters with which to change sampling interval, the time over which the function is defined, and other parameters. The default parameters are fine for now. Click on **OK** to generate the time-stamps. A .txt file appears in the *timeStamps* folder in the .dlim file's Eclipse project.

1.4.4 Extracting a DLIM Sequence from a Trace

Next we are going to extract a *Sequence* from an existing arrival rate trace. For this we use an arrival rate trace from the German Wikipedia. Download it here:

http://se2.informatik.uni-wuerzburg.de/files/wikipedia_trace.txt

The extraction process takes a DLIM *Sequence* and fills it with model elements modeling the arrival rates defined in the trace. **Right-click on the model's root** Sequence \rightarrow **Extract Sequence from Arrival Rate File** (Fig. 1.14).

· A Secu	n:/resource/LIMBO_Tutorial_Project/exampleMo rence exampleModel	Jaelaim
⊿ 💠 Sequ	New Child	
4 - 4	⁹ Undo Set	Ctrl+Z
5	Redo	Ctrl+Y
of	Cut	
D.		
⊳ - ≫	Delete	
	Validate	
~	Live Validation	
	Control	
Selection F	Debug As	
	Run As	
Propertie	Team	
Property	Compare With	
Name	Replace With	
Termina		
Termina	Load Resource	
	Refresh	
<	Show Properties View	
<u>.</u> 2	Remove from Context	Ctrl+Alt+Shift+Down
	Show Plot View	
	Extract Sequence from Arrival Rate File	

Figure 1.14: Extract Sequence from Arrival Rate Trace.

Set the downloaded trace as the Arrival Rate File, then select the Simple Process Extractor and click OK. The Periodic Process Extractor is explained in Section 1.5.1. Other extractors may appear, provided by third party plug-ins.

In the following dialog, set the **Seasonal Period** to **24** (the trace features hourly samples) and set the **Seasonal Periods per Trend** to **1** (This setting affects trend segment length, just as it did in the model creation wizard). Click **OK** (Fig. 1.15).

9	Extract S	Sequenc	e	×
Extract Sequence fro	m Arrival Rate File			
Seasonal Period: 24.0	Seasonal Periods p	er Trend:	1	
Select Seasonal Shape:	SinTrend	~		
Select Trend Shape:	SinTrend	~		
Select Trend Operator:	MULT	~		
?			OK Canc	el

Figure 1.15: Extract Sequence from Arrival Rate Trace.

1.4.5 Comparing Model and Trace

There are two ways to compare the extracted model instance to the original trace:

In the Plot View: Right-click \rightarrow **Toggle Arrival Rate File Plot** \rightarrow select the Wikipedia trace \rightarrow **OK**. You might want to also **Toggle Decomposition** again for better visibility. The Plot View now displays the arrival rates from the trace and the arrival rates of the model for comparison (Fig. 1.16).

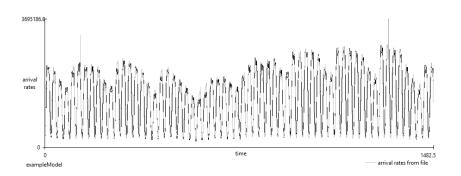


Figure 1.16: Plot View Comparison of Model and Trace.

Save the .dlim file $(\mathbf{ctrl}+\mathbf{s})$. In the Project Explorer Right-click on the .dlim file \rightarrow Calculate Difference to Arrival Rate File \rightarrow select the Wikipedia trace for the Arrival Rate File \rightarrow OK (Fig. 1.17). A dialog with a number of difference metrics appears. A list of all absolute differences is also written to the Eclipse project's *diffs* folder.

Project Ex ☆ ■ ↓ ↓ DiameStam ↓ example ↓ wikiped		MBO_Tutorial_Project/example Model
E withet	Open Open With	F3
		Ctrl+C Ctrl+V
3		Delete Ctrl+Alt+Shift+Down
	Mark as Landmark	Ctrl+Alt+Shift+Up
	Rename	F2
2		
ê	Refresh	F5
	Debug As Run As Team	> >
<	Compare With Replace With	*
	Properties	Alt+Enter
🖉 Tasks 🔷 Arriv	Calculate Difference to Arrival Rate File	
3695186.0	Generate Time Stamps	

Figure 1.17: Calculate Difference between Model and Trace.

1.5 Additional Features

These additional features are not part of the tutorial, but warrant additional explanation.

1.5.1 Periodic Process Extractor

The periodic extractor is a more complex extractor, which assumes that trends are repeating. For this the periodic extractor takes trend-segment-lists, which repeat. While the process allows for trend lists of arbitrary length, the GUI only allows for lists with 2 trend segments.

These lists can be added by filling the two text-fields below the list view in the extractor's dialog and then clicking **Add**.

Common inputs are weekly repeating trend lists with a total duration of 7 seasonal periods (days) (e.g.: 3,4) or monthly / 4-week lists with a total duration of 28 days (e.g.: 14,14) (Fig. 1.18).

Sequences derived using the Periodic Process Extractor are usually less accurate than Sequences derived using the Simple Process Extractor. They can however extend infinitely since their trends repeat.

a	Extract Sequen	ce	×
Extract Sequence from	n Arrival Rate File		
Seasonal Period: 24.0			
Piece-wise Trend per Se 3,4	asonais.		
Add Trend with followi	ng piece-wise durations: 14	14	Add
Select Seasonal Shape:	SinTrend 🗸		
Select Trend Shape:	SinTrend 🗸		
Select Trend Operator:	MULT 🗸		
?		ОК	Cancel

Figure 1.18: Periodic Process Extraction Dialog.

1.5.2 Difference Calculator

The difference calculator calculates the difference between an arrival rate trace file and a DLIM instance. In the Project Explorer Right-click on the .dlim file \rightarrow Calculate Difference to Arrival Rate File \rightarrow select the arrival rate trace for the Arrival Rate File \rightarrow OK. A list of all absolute differences is also written to the Eclipse project's *diffs* folder.

Additionally the difference calculator displays its results in a dialog (Fig. 1.19). This dialog displays the absolute and relative median and mean differences, as well as a relative curve difference based on the Dynamic Time Warping (DTW) algorithm. The DTW difference takes into account that the arrival rate variations may contain accurate arrival rates, yet be offset by time. As a result it is usually smaller than the relative mean and median differences. It is useful for comparing different model instances on the basis of the same trace.

Calculate Difference				
Difference Calculation Results				
Mean Difference:	1176407.873382452	, 8.073588502003048	%	
Median Difference:	1080277.3284446094	, 7.83752206442854	%	
Normalized Curve Difference (based on DTW):	0.017130930444837947			
?		ОК		

Figure 1.19: Results of a difference calculation.

Note: The difference calculator can only compare DLIM instances with traces containing tuples of time stamps and arrival rates. Running the difference calculator against pure time stamp lists results in an error. LIMBO does, however, include a feature to create

time stamp / arrival rate files from simple request time stamp lists. To do so, simply right click on your respective time stamp file and then select **Generate Arrival Rates from Time Stamps**.

1.6 Example Models

An Eclipse project with example Models can be downloaded from GitHub at:

https://github.com/joakimkistowski/LIMBO/tree/master/DLIM_examples