INTERNATIONAL STANDARD



First edition 2021-11

Systems and software engineering — High-level Petri nets —

Part 3: Extensions and structuring mechanisms

iTeh STIngénierie du logiciel et des systèmes → Réseaux de Petri de haut niveau — (standards iteh ai) Partie 3: Extensions et mécanismes de structuration

<u>ISO/IEC 15909-3:2021</u> https://standards.iteh.ai/catalog/standards/sist/351143f9-68ff-45bd-8137-1ca33b6f525e/iso-iec-15909-3-2021



Reference number ISO/IEC 15909-3:2021(E)

iTeh STANDARD PREVIEW (standards.iteh.ai)

<u>ISO/IEC 15909-3:2021</u> https://standards.iteh.ai/catalog/standards/sist/351143f9-68ff-45bd-8137-1ca33b6f525e/iso-iec-15909-3-2021



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2021

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office CP 401 • Ch. de Blandonnet 8 CH-1214 Vernier, Geneva Phone: +41 22 749 01 11 Email: copyright@iso.org Website: www.iso.org

Published in Switzerland

Contents

Introduction vi 1 Scope 1 2 Normative references 1 3 Terms and definitions 2 4 Conformance 2 4.1 General 2 4.2 Enrichment process 2 4.3 Extension process 2 4.4 Structuring mechanism 3 5.2 Instances of enrichment 3 5.2.1 General 3 5.2.2 Instances of enrichment 3 5.2.3 Reset arcs 4 5.2.4 Rend arcs 4 5.2.5 S.3.1 General 3 5.2.1 Reheral 3 5 5.3 Generalized curichment process 4 5.2.5 5.3 Definition of Patri nets with enrichment 6 5.3.3 Definition of Patri nets with enrichment 6 5.3.4 Filtering function for enrichment 6 5.3.5 Firing rule for Petri nets with enrichment 6 6.1 General 6 6.2.1	Forev	word		v	
2 Normative references 1 3 Terms and definitions 2 4 Conformance 2 4.1 General 2 4.2 Enrichment process 2 4.3 Extension process 2 4.4 Structuring mechanism 3 5 Enrichment process 3 5.1 General 3 5.2.1 Instances of enrichment 3 5.2.2 Inhibitor arcs 3 5.2.3 Reset arcs 4 5.2.4 Read arcs 4 5.2.5 S.3 General 3 5.3.1 General 7 5 5.3.2 Definition of enviones with envionent 5 5.3.3 Definition of enviones with envionent 6 5.3.4 Filtering function for envionent 6 5.3.5 Fring rule for Petri nets with envionent 6 5.3.6 Formalized envionent extensions SI 1430-654-654-654-654-654-654-654-654-654-654	Intro	ductior	1	vi	
3 Terms and definitions 2 4 Conformance 2 4.1 General 2 4.2 Enrichment process 2 4.3 Extension process 2 4.4 Structuring mechanism 3 5 Enrichment process 3 5.1 General 3 5.2.1 Instances of enrichment 3 5.2.2 Inhibitor arcs 3 5.2.3 Reset arcs 4 5.2.4 Read arcs 4 5.2.5 Capacity places 5 5.3.1 General 5 5.3.2 Definition of Periodity with envicionent 5 5.3.3 Definition of enabling rule for Pert nets with envicionent 6 5.3.4 Filtering function for enrichment 6 5.3.5 Fring rule for Petri nets with envicionent 6 5.3.6 Compatibility with extensionset \$311439-664-510-8137- 6 6 Extension process 6 6 6.1 General 6 6 6.2.1 Genera	1	Scope			
3 Terms and definitions 2 4 Conformance 2 4.1 General 2 4.2 Enrichment process 2 4.3 Extension process 2 4.4 Structuring mechanism 3 5 Enrichment process 3 5.1 General 3 5.2.1 Instances of enrichment 3 5.2.2 Inhibitor arcs 3 5.2.3 Reset arcs 4 5.2.4 Read arcs 4 5.2.5 Capacity places 5 5.3.1 General 5 5.3.2 Definition of Periodity with envicionent 5 5.3.3 Definition of enabling rule for Pert nets with envicionent 6 5.3.4 Filtering function for enrichment 6 5.3.5 Fring rule for Petri nets with envicionent 6 5.3.6 Compatibility with extensionset \$311439-664-510-8137- 6 6 Extension process 6 6 6.1 General 6 6 6.2.1 Genera	2	Norm	ative references	1	
4 Conformance 2 4.1 General 2 4.2 Enrichment process 2 4.3 Extension process 2 4.4 Structuring mechanism 3 5 Enrichment process 3 5.1 General 3 5.2.1 Instances of enrichment 3 5.2.1 General 3 5.2.2 Inhibitor arcs 3 5.2.3 Reset arcs 4 5.2.4 Read arcs 4 5.2.5 Generalized enrichment process IND PREVIEW 5 5.3.1 Generalized enrichment process IND PREVIEW 5 5.3.2 Definition of Petrinets with enrichment 6 5.3.3 Definition of Petrinets with enrichment 6 5.3.4 Filtering function for enrichment 6 5.3.5 Instance of extension: FIPO nets 6 6.1 General 6 6.2.4 Definition of FIPO nets 7 6.3.3 Definition of equivalent high-level Petri net 7 6.3.4 Definition of eq	_				
4.1 General 2 4.2 Enrichment process 2 4.3 Structuring mechanism 3 5 Enrichment process 3 5.1 General 3 5.2 Instances of enrichment 3 5.2.2 Inhibitor arcs 3 5.2.3 Reset arcs 4 5.2.4 Read arcs 4 5.2.5 Capacity places 5 5.3.1 Generalized enrichment process RD PREVIEW 5 5.3.2 Definition of ensity with enrichment 6 5.3.3 Definition of ensity with enrichment 6 5.3.3 Definition of ensity with enrichment 6 5.3.4 Filtering function for enrichment 6 5.3.5 Filtering function for enrichment 6 5.3.6 Comeral 6 6.1 General 6 6.2.1 General 6 6.2.2 Definition of FIPO nets 7 6.3.3 Definition of the envision structure 7 6.3.4 Definition of the envision process	_				
4.2 Enrichment process 2 4.3 Extension process 2 4.4 Structuring mechanism 3 5 Enrichment process 3 5.1 General 3 5.2.1 Instances of enrichment 3 5.2.2 Instances of enrichment 3 5.2.3 Reset arcs 3 5.2.4 Read arcs 4 5.2.5 Capacity places 5 5.3 Generalized enrichment process RD PREVIEW 5 5.3.1 Generalized enrichment process RD PREVIEW 5 5.3.3 Definition of Petri nets with enrichment 6 5.3.4 Filtering function for enrichment 6 5.3.5 Firing rule for Petri nets with enrichment 6 5.3.6 "Compatibility with enrichment 6 5.3.6 "Compatibility with enrichment 6 6.1 General 6 6.2.1 General 6 6.2.2 Definition of FIPO nets 7 6.3.3 Definition of FIPO nets 7 6.3.4 <t< th=""><th>т</th><th></th><th></th><th></th></t<>	т				
4.4 Structuring mechanism 3 5 Enrichment process 3 5.1 General 3 5.2 Instances of enrichment 3 5.2.1 General 3 5.2.2 Inbibitor arcs 3 5.2.3 Reset arcs 4 5.2.4 Read arcs 4 5.2.5 Generalized enrichment process RD PREVIEW 5 5.3.1 Generalized enrichment process RD PREVIEW 5 5.3.2 Definition of Petri nets with enrichment 6 5.3.3 Definition of Petri nets with enrichment 6 5.3.4 Filtering function for enrichment 6 5.3.5 General 6 6.3.4 Filtering rule for Petri nets with enrichment 6 5.3.6 "Instance of extension: FIFO nets 6 6.1 General 6 6.2.1 General 6 6.2.2 Definition of FIFO nets 7 6.3.3 Behavioural semantics 7 6.3.4 General 7 6.3.5 General		4.2			
5		4.3	1		
5.1 General 3 5.2 Instances of enrichment 3 5.2.1 General 3 5.2.2 Inhibitor arcs 3 5.2.3 Reset arcs 4 5.2.4 Read arcs 4 5.2.5 Capacity places 5 5.3 Generalized enrichment process RD PREVIEW 5 5.3.1 General 5 5.3.2 Definition of Petri nets with enrichment 6 5.3.3 Definition of renrichment 6 5.3.4 Filtering function for enrichment 6 5.3.5 Firing rule for Petri nets with enrichment 6 5.3.6 Compatibility with extraions statistical 149-68145bd-8137- 6 6.3 Firing rule for Petri nets 6 6.1 General 6 6.2 An instance of extension: FIFO nets 6 6.2.1 General 7 6.2.2 Definition of equivalent high-level Petri net 7 6.3.3 Definition of the net type 6 6.3 Definition of the net type 6		4.4	Structuring mechanism		
5.2 Instances of enrichment 3 5.2.1 General 3 5.2.2 Inbibitor arcs 3 5.2.3 Reset arcs 4 5.2.4 Read arcs 4 5.2.5 Generalized enrichment process. FD PREVIEW 5.3.1 Generalized enrichment process. FD PREVIEW 5.3.2 Definition of Petri nets with enrichment 6 5.3.3 Definition of or enrichment 6 5.3.3 Definition of or enrichment 6 5.3.4 Filtering function for enrichment 6 5.3.5 Definition of Petri nets with enrichment 6 5.3.6 Incompatibility with extensions $st(35)(14)9-681-450-8137-$ 6 6.1 General 6 6.2.4 Instance of extension: FIFO nets 6 6.1 General 6 6.2.2 Definition of FIFO nets 7 6.3.4 Behavioural semantics 7 6.3.5 Definition of the net type 8 6.3.1 General 7 6.3.2 Definition of the net type 8 <td>5</td> <td colspan="4">Enrichment process</td>	5	Enrichment process			
		-			
5.2.2Inhibitor arcs35.2.3Reset arcs45.2.4Read arcs45.2.5Capacity places55.3Generalized enrichment process FD PREVIEW55.3.1General55.3.2Definition of Petrinets with enrichment65.3.3Definition of enabling rule for Petri nets with enrichment65.3.4Filtering function for enrichment65.3.5Firing rule for Petri nets with enrichment65.3.6"Compatibility with extensions" $5311439-68145bd-8137$ 66Extension process66.1General66.2An instance of extension: FIFO nets66.2.1General66.2.2Definition of FIFO nets66.2.3Definition of FIFO nets76.2.4Definition of FIFO nets76.3.1General76.3.2Definition of the net type86.3.3Definition of the net type86.3.3Definition of the behavioural semantics87.1General87.2Definition of module interface97.3Module definition97.3Definition of module instances and uses97.3Definition of module instances and uses97.3Definition of module instances and uses97.3Definition of module lenstances and uses97.3Definition of module lenstances and uses9 </td <td></td> <td>5.2</td> <td></td> <td></td>		5.2			
5.2.3Reset arcs45.2.4Read arcs45.2.5Capacity places55.3Generalized enrichment process RD PREVIEW55.3.1General55.3.2Definition of Petri nets with enrichment55.3.3Definition of reading rule for Petri nets with enrichment65.3.4Filtering function for enrichment65.3.5Firing rule for Petri nets with enrichment65.3.6Compatibility with extensions existil 1439-688-4504-813766Extension process66.1General66.2An instance of extension: FIFO nets66.3The generalized extension process76.2.3Behavioural semantics76.2.4Definition of FIFO nets76.2.5Compatibility with enrichments76.3The generalized extension process76.3.1General76.3.3Definition of the net type87.3Module definition87.1General87.2Definition of module inplementation97.3Definition of module instances and uses97.3Definition of module instances and uses97.3					
5.2.4Read arcs45.2.5Capacity places55.3Generalized enrichment process RD PREVIEW55.3.1Definition of Petrinets with enrichment65.3.2Definition of enrichment65.3.3Definition of enrichment65.3.4Filtering function for enrichment65.3.5Firing rule for Petri hets with enrichment65.3.6in Compatibility with extensions/511439-68ff-45de-813766Extension process66.1General66.2An instance of extension: FIFO nets66.3.1General66.2.2Definition of FIFO nets76.2.3Behavioural semantics76.2.4Definition of FIFO nets76.3.7He generalized extension process76.3.1General76.3The generalized extension process76.3.1General76.3.1General76.3.2Definition of the net type86.3.3Definition of the net type86.3.3Definition of module interface97.2.1Definition of module interface97.3Module definition97.3Definition of module instances and uses97.3.3Definition of module instances and uses97.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of signature and homomorphisms σ_k 11					
5.2.5Capacity places55.3Generalized enrichment process RD PREVIEW55.3.1General55.3.2Definition of Petri nets with enrichment55.3.3Definition of enabling rule for Petri nets with enrichment65.3.4Filtering function for enrichment65.3.5Firing rule for Petri nets with enrichment65.3.6In Compatibility with extensions st/3511439-688-45bd-8137-66Extension process66.1General66.2.1General66.2.2Definition of FIFO nets66.3.3Definition of FIFO nets76.4.4Definition of FIFO nets76.5.5Compatibility with enrichments76.2.1General66.2.2Definition of FIFO nets76.3.3Behavioural semantics76.3The generalized extension process76.3.1General76.3.3Definition of the net type86.3.3Definition of the behavioural semantics87Structuring mechanism87.2Module definition97.3Module instantiation97.3Definition of module interface97.3Definition of module instances and uses97.3Definition of module instances and uses97.3Definition of module instances and uses97.3Definition of module definition					
5.3General content process RD PREVIEW55.3.1General55.3.2Definition of Patri nets with enrichment55.3.3Definition of enabling rule for Petri nets with enrichment65.3.4Filtering function for enrichment65.3.5Firing rule for Petri nets with enrichment65.3.6Compatibility with extensions: Solid 30-68ft 45bd-8137-66Extension process66.1General66.2.2Definition of FIFO nets66.2.3Behavioural semantics76.2.4Definition of FIFO nets76.2.5Compatibility with enrichments76.2.6General66.2.1General66.2.2Definition of FIFO nets76.3.3Behavioural semantics76.3.4Definition of the net type86.3.3Definition of the net type86.3.3Definition of the net type87.4Module definition87.2.1Definition of module interface97.3.3Definition of module interface97.3.4Definition of module instances and uses97.3.4Definition of module definition107.3.4Definition of variables11					
5.3.2Definition of Petri nets with enrichment55.3.3Definition of enabling rule for Petri nets with enrichment65.3.4Filtering function for enrichment65.3.5Firing rule for Petri nets with enrichment65.3.6Compatibility with extensions $(3511439-6814504-8137)$ 66Extension process66.1General66.2An instance of extension: FIFO nets66.3.3Definition of FIFO nets76.2.4Definition of equivalent high-level Petri net76.3.5Compatibility with enrichments76.3The generalized extension process76.3.1General676.3.2Definition of the net type86.3Definition of the behavioural semantics87Structuring mechanism87.1General87.2.1Definition of sort generator87.3Module instantiation97.3Definition of module implementation97.3Definition of module instances and uses97.3Definition of module instances and uses97.3Definition of module instances and uses97.3Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11		52	5.2.5 Capacity places		
5.3.2Definition of Petri nets with enrichment55.3.3Definition of enabling rule for Petri nets with enrichment65.3.4Filtering function for enrichment65.3.5Firing rule for Petri nets with enrichment65.3.6Compatibility with extensions511439-681-4564-81376Extension process66.1General66.2An instance of extension: FIFO nets66.3.3Behavioural semantics76.2.4Definition of FIFO nets76.2.3Behavioural semantics76.3.4Definition of equivalent high-level Petri net76.3.3Definition of the net type86.3The generalized extension process76.3.1General76.3.2Definition of the behavioural semantics87Structuring mechanism87.1General87.2.5Definition of sort generator87.3Module instantiation97.3.1General97.3.2Definition of module instances and uses97.3.4Definition of module definition107.3.4Definition of signature and homomorphisms σ_k 11		5.5	5.3.1 General Standard PREVIEW		
5.3.3Definition of enabling rule for Petri nets with enrichment65.3.4Filtering function for enrichment65.3.5Firing rule for Petri nets with enrichment65.3.6Compatibility with extensions with 51143D-68ff-45bd-8137-66Extension process66.1General66.2.2Definition of FIFO nets66.2.3Behavioural semantics76.2.4Definition of FIFO nets76.2.5Compatibility with enrichments76.3The generalized extension process76.3.1General76.3.2Definition of the net type86.3.3Definition of the behavioural semantics76.3.4Definition of sort generator87.4General87.2Module definition97.3Module instantiation97.3.3Definition of module instances and uses97.3.3Definition of module instances and uses97.3.4Definition of sort ganerator117.3.5Definition of sort and uses11			5.3.2 Definition of Petri nets with enrichment		
5.3.4Filtering function for enrichment65.3.5Firing rule for Petri nets with enrichment65.3.6Compatibility with extensions $sit(35114)$ (9-68145bd-8137-61ca/3b052erso-ec-15909-3-202166Extension process66.1General66.2An instance of extension: FIFO nets66.2.1General66.2.2Definition of FIFO nets76.2.3Behavioural semantics76.2.4Definition of equivalent high-level Petri net76.2.5Compatibility with enrichments76.3The generalized extension process76.3Definition of the net type86.3.3Definition of the net type86.3.3Definition of sort generator87.1General87.2.1Definition of sort generator87.2.2Definition of module interface97.3.3Definition of module interface97.3.4Definition of module instances and uses97.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of signature and homomorphisms σ_k 11					
$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$			5.3.4 Filtering function for enrichment	6	
6Extension process66.1General66.2An instance of extension: FIFO nets66.2.1General66.2.2Definition of FIFO nets76.2.3Behavioural semantics76.2.4Definition of equivalent high-level Petri net76.2.5Compatibility with enrichments76.3The generalized extension process76.3.1General76.3.2Definition of the net type86.3.3Definition of the behavioural semantics87Structuring mechanism87.1General87.2Module definition87.2.3Definition of module interface97.3.1General97.3.2Definition of module instances and uses97.3.3Definition of module definition107.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11			5.3.5 Firing rule for Petri nets with enrichment	6	
6Extension process66.1General66.2An instance of extension: FIFO nets66.2.1General66.2.2Definition of FIFO nets76.2.3Behavioural semantics76.2.4Definition of equivalent high-level Petri net76.2.5Compatibility with enrichments76.3The generalized extension process76.3.1General76.3.2Definition of the net type86.3.3Definition of the behavioural semantics87Structuring mechanism87.1General87.2Module definition87.2.3Definition of module interface97.3.1General97.3.2Definition of module instances and uses97.3.3Definition of module definition107.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11			5.3.6 http://tandetibility.withlex/tandard/sist/35114319-68ff-45bd-8137-	6	
6.1General66.2An instance of extension: FIFO nets66.2.1General66.2.2Definition of FIFO nets76.2.3Behavioural semantics76.2.4Definition of equivalent high-level Petri net76.2.5Compatibility with enrichments76.3The generalized extension process76.3.1General76.3.2Definition of the net type86.3.3Definition of the behavioural semantics87Structuring mechanism87.1General87.2.1Definition of sort generator87.2.2Definition of module interface97.3.1General97.3.2Definition of module instances and uses97.3.3Definition of module instances and uses97.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11	6	Extension process			
6.2.1General66.2.2Definition of FIFO nets76.2.3Behavioural semantics76.2.4Definition of equivalent high-level Petri net76.2.5Compatibility with enrichments76.3The generalized extension process76.3.1General76.3.2Definition of the net type86.3.3Definition of the behavioural semantics87Structuring mechanism87.1General87.2Module definition87.2.1Definition of sort generator87.2.2Definition of module interface97.3.3Definition of module instances and uses97.3.4Definition of module definition107.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11	-				
6.2.2Definition of FIFO nets76.2.3Behavioural semantics76.2.4Definition of equivalent high-level Petri net76.2.5Compatibility with enrichments76.3The generalized extension process76.3.1General76.3.2Definition of the net type86.3.3Definition of the behavioural semantics87Structuring mechanism87.1General87.2Module definition87.2.1Definition of sort generator87.2.3Definition of module interface97.2.3Definition of module implementation97.3Module instantiation97.3.1General97.3.2Definition of module instances and uses97.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11		6.2	An instance of extension: FIFO nets	6	
6.2.3Behavioural semantics76.2.4Definition of equivalent high-level Petri net76.2.5Compatibility with enrichments76.3The generalized extension process76.3.1General76.3.2Definition of the net type86.3.3Definition of the behavioural semantics87Structuring mechanism87.1General87.2Module definition87.2.1Definition of sort generator87.2.2Definition of module interface97.3.3Definition of module instances and uses97.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11			6.2.1 General	6	
6.2.4Definition of equivalent high-level Petri net76.2.5Compatibility with enrichments76.3The generalized extension process76.3.1General76.3.2Definition of the net type86.3.3Definition of the behavioural semantics87Structuring mechanism87.1General87.2Module definition87.2.1Definition of sort generator87.2.2Definition of module interface97.2.3Definition of module implementation97.3Module instantiation97.3.1General97.3.2Definition of module instances and uses97.3.3Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11			6.2.2 Definition of FIFO nets	7	
6.2.5Compatibility with enrichments76.3The generalized extension process76.3.1General76.3.2Definition of the net type86.3.3Definition of the behavioural semantics87Structuring mechanism87.1General87.2Module definition87.2.1Definition of sort generator87.2.2Definition of module interface97.2.3Definition of module implementation97.3Module instantiation97.3.1General97.3.2Definition of module instances and uses97.3.3Definition of module definition107.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11					
6.3The generalized extension process76.3.1General76.3.2Definition of the net type86.3.3Definition of the behavioural semantics87Structuring mechanism87.1General87.2Module definition87.2.1Definition of sort generator87.2.2Definition of module interface97.2.3Definition of module implementation97.3Module instantiation97.3.1General97.3.2Definition of module instances and uses97.3.3Definition of module definition107.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11					
6.3.1General76.3.2Definition of the net type86.3.3Definition of the behavioural semantics87Structuring mechanism87.1General87.2Module definition87.2Definition of sort generator87.2.1Definition of module interface97.2.3Definition of module implementation97.3Module instantiation97.3.1General97.3.2Definition of module instances and uses97.3.3Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11			6.2.5 Compatibility with enrichments		
6.3.2Definition of the net type86.3.3Definition of the behavioural semantics87Structuring mechanism87.1General87.2Module definition87.2Definition of sort generator87.2.1Definition of module interface97.2.2Definition of module implementation97.3Module instantiation97.3Definition of module instances and uses97.3.1General97.3.2Definition of module definition107.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11		6.3			
6.3.3 Definition of the behavioural semantics87Structuring mechanism87.1 General87.2 Module definition87.2 Module definition of sort generator87.2.1 Definition of sort generator87.2.2 Definition of module interface97.2.3 Definition of module implementation97.3 Module instantiation97.3.1 General97.3.2 Definition of module instances and uses97.3.3 Definition of module definition107.3.4 Definition of signature and homomorphisms σ_k 117.3.5 Definition of variables11					
7Structuring mechanism87.1General87.2Module definition87.2.1Definition of sort generator87.2.2Definition of module interface97.2.3Definition of module implementation97.3Module instantiation97.3.1General97.3.2Definition of module instances and uses97.3.3Definition of module definition107.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11					
7.1General87.2Module definition87.2.1Definition of sort generator87.2.2Definition of module interface97.2.3Definition of module implementation97.3Module instantiation97.3.1General97.3.2Definition of module instances and uses97.3.3Definition of module definition107.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11	_	_			
7.2Module definition87.2.1Definition of sort generator87.2.2Definition of module interface97.2.3Definition of module implementation97.3Module instantiation97.3.1General97.3.2Definition of module instances and uses97.3.3Definition of module definition107.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11	7				
7.2.1Definition of sort generator87.2.2Definition of module interface97.2.3Definition of module implementation97.3Module instantiation97.3.1General97.3.2Definition of module instances and uses97.3.3Definition of module definition107.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11					
7.2.2Definition of module interface97.2.3Definition of module implementation97.3Module instantiation97.3General97.3.2Definition of module instances and uses97.3.3Definition of module definition107.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11		1.2			
7.2.3 Definition of module implementation97.3 Module instantiation97.3.1 General97.3.2 Definition of module instances and uses97.3.3 Definition of module definition107.3.4 Definition of signature and homomorphisms σ_k 117.3.5 Definition of variables11					
7.3Module instantiation97.3.1General97.3.2Definition of module instances and uses97.3.3Definition of module definition107.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11					
7.3.1General97.3.2Definition of module instances and uses97.3.3Definition of module definition107.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11		7.3			
7.3.2Definition of module instances and uses97.3.3Definition of module definition107.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11					
7.3.3Definition of module definition107.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11					
7.3.4Definition of signature and homomorphisms σ_k 117.3.5Definition of variables11					
7.3.5 Definition of variables			734 Definition of signature and homomorphisms σ	11	
u - ·					

Bibliography		15
Annex A (informativ	ve) Guidelines for graphical notations	13
7.3.7	Definition of places and transitions \hat{T}	12

iTeh STANDARD PREVIEW (standards.iteh.ai)

<u>ISO/IEC 15909-3:2021</u> https://standards.iteh.ai/catalog/standards/sist/351143f9-68ff-45bd-8137-1ca33b6f525e/iso-iec-15909-3-2021

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iso.org/directiv

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see <u>www.iso.org/patents</u>) or the IEC list of patent declarations received (see <u>mww.iso.org/patents</u>) or the IEC list of patent declarations received.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC⁸JTC 1, Information technology, Subcommittee SC 7, Software and systems engineering.⁰⁹⁻³⁻²⁰²¹

A list of all parts in the ISO/IEC 15909 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at <u>www.iso.org/members.html</u> and <u>www.iec.ch/national-committees</u>.

Introduction

Petri nets have been used to describe a wide range of systems since their invention in 1962. The technique is mathematically defined and can thus be used to provide unambiguous specifications and descriptions of applications. It is also an executable technique, allowing specification prototypes to be developed to test ideas at the earliest and cheapest opportunity. Specifications written in the technique can be subjected to analysis methods to prove properties about the specifications, before implementation commences, thus saving on testing and maintenance time and providing a high level of quality assurance.

A problem with Petri nets is the explosion of the number of elements in their graphical form when they are used to describe complex systems. High-level Petri nets were developed to overcome this problem by introducing higher level concepts, such as the use of complex structured data as tokens, and using algebraic expressions to annotate net elements. The use of "high-level" to describe these Petri nets is analogous to the use of "high-level" in high-level programming languages (as opposed to assembly languages), and is the usual term used in the Petri net community. Two of the early forms of high-level nets that this document builds on are predicate-transition nets and coloured Petri nets, first introduced in 1979 and developed during the 1980s. It also uses some of the notions developed for algebraic Petri nets, first introduced in the mid-1980s. It is believed that this document captures the spirit of these earlier developments (see Bibliography).

The technique has multiple uses. For example, it can be used directly to specify systems or to define the semantics of other less formal languages. It can also serve to integrate techniques currently used independently such as state-transition diagrams and data flow diagrams. The technique is particularly suited to parallel and distributed systems development as it supports concurrency. The technique is able to specify systems at a level that is independent of the choice of implementation (i.e. by software, hardware (electronic and/or mechanical) or humans or a combination). This document may be cited in contracts for the development of systems (particularly distributed systems) or used by application developers or Petri net tool vendors or users. ISO/IEC 15909-3:2021

https://standards.iteh.ai/catalog/standards/sist/351143f9-68ff-45bd-8137-The ISO/IEC 15909 series is concerned with defining a modelling language and its transfer format, known as high-level Petri nets. ISO/IEC 15909-1 provides the mathematical definition of high-level Petri nets, called the semantic model, the graphical form of the technique, known as high-level Petri net graphs (HLPNGs), and its mapping to the semantic model. It also introduces some common notational conventions for HLPNGs.

ISO/IEC 15909-2 defines a transfer format for high-level Petri nets in order to support the exchange of high-level Petri nets among different tools. This format is called the Petri net markup language (PNML). Since there are many different types of Petri nets in addition to high-level Petri nets, ISO/IEC 15909-2 defines the core concepts of Petri nets along with an XML syntax, which can be used for exchanging any kind of Petri nets. Based on this PNML core model, ISO/IEC 15909-2 also defines the transfer syntax for the types of Petri nets that are defined in ISO/IEC 15909-1: place/transition nets, symmetric nets¹), high-level Petri nets, Petri nets with priorities, and Petri nets with time. Place/transition nets and symmetric nets can be considered to be restricted versions of high-level Petri nets. Petri nets with priorities and Petri nets with time are considered as extensions of the other types.

This document defines extensions to the types of Petri nets that are defined in ISO/IEC 15909-1. These extensions comprise enrichments of Petri net types and definitions of new Petri net types. This document also defines structuring mechanisms for these Petri net types.

In this document, the semantics which is considered is always the interleaving semantics.

This document provides an abstract mathematical syntax and a formal semantics for the technique. Conformance to the document is possible at several levels. The level of conformance depends on the class of high-level net chosen. The usual graphical notations are depicted in Annex A.

¹⁾ Symmetric nets have been first introduced as well-formed nets and are currently standardized as ISO/IEC 15909-1.

Systems and software engineering — High-level Petri nets —

Part 3: Extensions and structuring mechanisms

1 Scope

This document defines enrichments, extensions and structuring mechanisms of Petri nets, applied on the definitions proposed in ISO/IEC 15909-1. This document facilitates the definitions of new kinds of Petri nets and their interoperability, while remaining compatible with those defined in ISO/IEC 15909-1.

This document is written as a reference for designers of new Petri net variants, by defining common enrichments, extensions and structuring mechanisms, as well as a generalized process for defining new ones.

This document is applicable to a wide variety of concurrent discrete event systems and in particular distributed systems. Generic fields of application include: **REVIEW**

requirements analysis;

(standards.iteh.ai)

development of specifications, designs and test suites;

— ISO/IEC 15909-32021
— descriptions of existing systems prior to re-engineering;
— https://standards.itel.arcatalog/standards/sist/3511499-68ff-45bd-8137-

- modelling business and software processes;
- providing the semantics for concurrent languages;
- simulation of systems to increase confidence;
- formal analysis of the behaviour of systems;
- and development of Petri net support tools.

This document can be applied to the design of a broad range of systems and processes, including aerospace, air traffic control, avionics, banking, biological and chemical processes, business processes, communication protocols, computer hardware architectures, control systems, databases, defence command and control systems, distributed computing, electronic commerce, fault-tolerant systems, games, hospital procedures, information systems, Internet protocols and applications, legal processes, logistics, manufacturing systems, metabolic processes, music, nuclear power systems, operating systems, transport systems (including railway control), security systems, telecommunications and workflow.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 15909-1, Systems and software engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 15909-1 and the following apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

- ISO Online browsing platform: available at https://www.iso.org/obp
- IEC Electropedia: available at <u>https://www.electropedia.org/</u>

3.1

capacity

maximum multiset of tokens a *capacity place* (3.2) can hold

3.2

capacity place

special kind of place that can hold no more than a specified *capacity* (3.1)

3.3

FIFO

special kind of queue that is operated first in first out

3.4

inhibitor arc

special kind of arc that reverses the logic of an input place

Note 1 to entry: Instead of testing the presence of some tokens in the related place, it tests the lack of these. (standards.iteh.ai)

3.5

read arc

special kind of arc that tests the presence of some tokens in the related place, without consumption https://standards.itel.ai/consumption/lca33b6f525e/iso-iec-15909-3-2021

3.6

reset arc special kind of arc that empties an input place

3.7

sort generator

generator of new sorts and operators built from a given signature (passed as a parameter)

4 Conformance

4.1 General

There are different levels of conformance to this document.

4.2 Enrichment process

A Petri net model is conformant to the enrichment process level when it contains a subset of the features defined as enrichments by <u>Clause 5</u>.

4.3 Extension process

A Petri net model is conformant to the extension process level when it contains a single feature defined as extension by <u>Clause 6</u>.

4.4 Structuring mechanism

A Petri net model is conformant to the structuring mechanism level when it embeds a structuring feature as defined by <u>Clause 7</u>.

5 Enrichment process

5.1 General

This clause defines enrichments of Petri nets. For each enrichment, it first provides a syntactic definition, then its individual semantics, and finally a formulation of the semantics allowing for composition with other enrichments or extensions.

5.2 Instances of enrichment

5.2.1 General

This subclause defines commonly used enrichments. The list provided here is not exhaustive. The general mechanism to define a new enrichment is provided in 5.3.

NOTE Enrichments are orthogonal one to another. Hence, they can be used in combination among themselves, and to enrich the different types of nets defined in this document.

5.2.2 Inhibitor arcs Teh STANDARD PREVIEW

5.2.2.1 Definition of Petri nets with inhibitor arcs

A Petri net with inhibitor arcs is a Petri net N together with a matrix $I \in AN^{|P| \times |T|}$ of inhibitor arcs. https://standards.iteh.ai/catalog/standards/sist/35114319-68ff-45bd-8137-

5.2.2.2 Definition of enabling rule for Petri nets with inhibitor arcs

A transition $t \in T$ is enabled in marking *M*, denoted by M[t), iff:

 $\forall p \in \bullet t, (M(p) \ge \operatorname{Pre}(p,t)) \land (I(p,t) = 0 \lor M(p) < I(p,t))$

5.2.2.3 Filtering function for inhibitor arcs

The filtering function F_i ($N \in N$, $t \in T_N$) returns true when there are less tokens than the value specified on the inhibitor arc ($I(p,t) \neq 0$).

$$F_{i}(N \in \mathcal{N}, t \in T_{N}) : \begin{cases} \text{True, iff } \forall p \in \bullet t : M(p) < I(p,t) \\ \text{False, otherwise} \end{cases}$$

The enabling function for place/transition nets with inhibitor arcs is thus:

 $E_{\text{pti}}(N,t) = E_{\text{pt}}(N,t) \wedge F_{\text{i}}(N,t)$

The enabling function for symmetric nets with inhibitor arcs is thus:

 $E_{\mathrm{sni}}(N,t) = E_{\mathrm{sn}}(N,t) \wedge F_{\mathrm{i}}(N,t)$

The enabling function for high-level Petri nets with inhibitor arcs is thus:

 $E_{\text{hlpni}}(N,t) = E_{\text{hlpn}}(N,t) \wedge F_{\text{i}}(N,t)$

ISO/IEC 15909-3:2021(E)

NOTE Multiple combinations of filtering functions can be defined (see <u>5.3</u> and <u>6.3</u>), e.g. for Petri nets with priorities and inhibitor arcs.

5.2.2.4 Firing rule for Petri nets with inhibitor arcs

The firing rule for Petri nets with inhibitor arcs is the same as for the corresponding Petri net type. Hence, A_{pre} and A_{post} are those of the corresponding Petri net type.

5.2.3 Reset arcs

5.2.3.1 Definition of Petri nets with reset arcs

A Petri net with reset arcs is a Petri net N together with a matrix $Rt \in \{0,1\}^{|P| \times |T|}$ of reset arcs.

5.2.3.2 Definition of enabling rule for Petri nets with reset arcs

A transition $t \in T$ is enabled in marking M, denoted by M[t), iff:

 $\forall p \in \bullet t, M(p) \ge \operatorname{Pre}(p,t)$

5.2.3.3 Filtering function for reset arcs

The filtering function F_{reset} ($N \in \mathcal{N}, t \in T_N$) returns true for all places.

5.2.3.4 Augmented firing rule for reset arcs dards.iteh.ai)

The firing rule for Petri nets with reset arcs relies on the one of the corresponding Petri net type. A_{pre} is the one of the corresponding Petri net type. A_{post} is the one of the corresponding Petri net type augmented by removing all tokens referenced in A_{N} from the connected place.

5.2.4 Read arcs

5.2.4.1 Definition of Petri nets with read arcs

A Petri net with read arcs is a Petri net N together with a matrix $Rd \in AN^{|P| \times |T|}$ of read arcs.

5.2.4.2 Definition of enabling rule for Petri nets with read arcs

A transition $t \in T$ is enabled in marking M, denoted by M[t), iff:

 $\forall p \in \bullet t, (M(p) \ge \operatorname{Pre}(p,t) \land M(p) \ge Rd(p,t)$

5.2.4.3 Filtering function for read arcs

The filtering function F_{read} ($N \in \mathcal{N}, t \in T_N$) is the same as the enabling function, applied to Rd.

5.2.4.4 Augmented firing rule for read arcs

The firing rule for Petri nets with read arcs is the same as for the corresponding Petri net type. Hence, A_{pre} and A_{post} are those of the corresponding Petri net type.

NOTE Tokens in read arc annotation Rd(p,t) are not consumed.

5.2.5 Capacity places

5.2.5.1 Definition of Petri nets with capacity places

A Petri net with capacity place is a Petri net N together with a vector $C \in (\mathbb{N} \cup \{\infty\})^{|P|}$ of place capacities.

5.2.5.2 Definition of enabling rule for Petri nets with capacity places

A transition $t \in T$ is enabled in marking *M*, denoted by M[t), iff:

 $[\forall p \in \bullet t : M(p) \ge \operatorname{Pre}(p,t)] \land [\forall p' \in t \bullet : M(p') + \operatorname{Post}(p',t) - \operatorname{Pre}(p',t) \le C(p')]$

NOTE For all Petri net types, only the number of tokens in a place is limited by its capacity.

5.2.5.3 Filtering function for capacity places

The filtering function $F_c(N \in \mathcal{N}, t \in T_N)$ returns true when the firing would not lead to exceeding the capacity for output places.

 $F_{c}(N \in \mathcal{N}, t \in T_{N}): \begin{cases} \text{True, iff } [\forall p \in \bullet t : M(p) \ge \Pr(p, t)] \land [\forall p' \in t \bullet : M(p') + \operatorname{Post}(p', t) - \operatorname{Pre}(p', t) \le C(p')] \\ \text{False, otherwise} \end{cases}$

The enabling function for place/transition nets with capacity places is thus:

$$E_{\text{ptc}}(N,t) = E_{\text{pt}}(N,t) \wedge F_{\text{c}}(N,t)$$
 (standards.iteh.ai)

The enabling function for symmetric nets with capacity places is thus:

 $E_{\rm snc}(N,t) = E_{\rm sn}(N,t) = E_{\rm sn}(N,t) + E_{\rm sn}(N,t) = E_{\rm sn}(N,t) + E_{\rm sn}(N,t) = E_{\rm sn}(N,t) + E_{\rm sn}(N,$

The enabling function for high-level Petri nets with capacity places is thus:

 $E_{\text{hlpnc}}(N,t) = E_{\text{hlpn}}(N,t) \wedge F_{\text{c}}(N,t)$

NOTE Multiple combinations of filtering functions can be defined (see <u>5.3</u> and <u>6.3</u>), e.g. for Petri nets with priorities and capacity places.

5.2.5.4 Firing rule for Petri nets with capacity place

The firing rule for Petri nets with capacity places is the same as for the corresponding Petri net type. Hence, A_{pre} and A_{post} are those of the corresponding Petri net type.

5.3 Generalized enrichment process

5.3.1 General

This subclause defines the template used to define a new enrichment to an existing Petri net type.

5.3.2 Definition of Petri nets with enrichment

A Petri net with ε is a Petri net N together with a definition D of the enrichment according to the net elements.