Function

From EPC Standard

The printable version is no longer supported and may have rendering errors. Please update your browser bookmarks and please use the default browser print function instead.

Function	
Graphical Notation	
	
IsSubClassOf	IsSubClassOf::Process element
Successors	hasSuccessor::Event, hasSuccessor::Operator
Predecessors	hasPredecessor::Event, hasPredecessor::AND Operator
HasIncomingControlFlow	hasIncomingControlFlow::1
HasOutgoingControlFlow	hasOutgoingControlFlow::1
HasResource	hasResource::0, hasResource::n
HasAttribute	hasAttribute::0, hasAttribute::n
Edit the Properties	

□

Contents

Brief Information

This is an autogenerated section!

You are not able to edit this information by hand, but by edit the Form (and therefore the properties) of

this page. Please refer to the Edit the properties link at the bottom of the info box. {{#show: Function | ?Is a | Intro=The Function is a }}. {{#show: Function | ?contains | Intro=It contains }}. {{#show: Function | ?hasSuccessor | Intro=Possible succeeding element(s) is/are }}. {{#show: Function | ?hasPredecessor | Intro=Previous element(s) can be }}. {{#show: Function | ?hasIncomingControlFlow | Intro=The cardinalities are | Outro= (incoming)}} {{#show: Function | ?hasOutgoingControlFlow | Intro=and | Outro= (outgoing) respectively }}. {{#show: Function | ?refersTo | Intro=The Function refers to }}. {{#show: Function | ?attachedTo | Intro=The Function is attached to a }}.

Syntax

The EPC syntax requires a function either to be preceded and followed by an operator or an event. Like events, functions may be linked to its predecessor and successor via a control flow arc. The control flow arc cannot connect two functions directly. [1], [2], [3] A function may be attached with additional resources such as an organizational unit, data object or information system, which add further detail to the process element.[4]

Semantics

Functions are triggered by events. They are the active elements of an EPC that represent the activities of a modelled process and again raise events upon completion.[5][6][7][8][9][10] The extended event driven process chain expands the semantic of functions, since they may be associated with the organizational unit performing the function or the data needed and created as output by the function. [8]

Semantic Representation

A function F is a part of an EPC = (E, F, P, C, l, A), for which F is defined:

An element of F is called function. $F \neq \emptyset$ F is a pairwise disjoint and finite set $E \cap F = \emptyset$, $F \cap C = \emptyset$ 1.

It is also called a node N, being part of $N = E \cup F \cup P \cup C$.[11]

Following requirements are made on functions so an EPC can be called relaxed syntactically correct:

- A function is connected to other nodes ($\bullet f$ and $f \bullet$) by incoming and outgoing arcs.
- Functions have exactly one incoming and one outgoing arc: For each $f \in F$: $|\bullet f| = 1 \wedge |f \bullet| = 1$.
- A function neither starts nor ends an EPC.

- $|F| \geq 1$. There is at least one function in an EPC.

- $\forall f \in F : \bullet f \subseteq (E \cup CEF) \wedge f \bullet \subseteq (E \cup CFE)$

Functions must have events or ef-connectors in the preset and events or fe-connectors in the postset. That means a function always follows an event, which may be linked via connectors (except for end events).[12], [13]

Pragmatic

Linguistic Correctness

To satisfy the requirements of pragmatic correctness every label of the model elements should follow a specified naming convention. A function represents the active and time consuming part of an EPC model, thereby a function’s name should reflect its characteristic as an activity and be created from a substantive and an active.[3] The English naming convention suggestions for functions differ from the

German conventions, as it is shown in the following table:

Language	Rule	Example
English	Verb + Substantive(s)	"Processing order"
German	Substantive(s) + Verb	"Bestellung bearbeiten"

The chosen name of a function should precisely describe, what is done at this specific part of the underlying process. To fulfill this requirement the naming process should be based heavily on the activities found in the process description.[5]

XML Representation

<xs:complexType name="typeFunction"> <xs:sequence> <xs:element name="documentation" type="xs:anyType" minOccurs="0"/> <xs:element name="toolInfo" type="xs:anyType" minOccurs="0"/> <xs:element name="name" type="xs:string"/> <xs:element name="description" type="xs:string" minOccurs="0"/> <xs:choice minOccurs="0"> <xs:element name="graphics" type="epml:typeGraphics"/> </xs:choice> <xs:choice minOccurs="0"> <xs:element name="syntaxInfo"> <xs:complexType> <xs:attribute name="implicitType"> <xs:simpleType> <xs:restriction base="xs:string"> <xs:enumeration value="function"/> </xs:restriction> </xs:simpleType> </xs:attribute> </xs:complexType> </xs:element> </xs:choice> <xs:choice minOccurs="0"> <xs:element name="toProcess" type="epml:typeToProcess"/> </xs:choice> <xs:choice minOccurs="0" maxOccurs="unbounded"> <xs:element name="attribute" type="epml:typeAttribute"/> </xs:choice> </xs:sequence> <xs:attribute name="id" type="xs:positiveInteger" use="required"/> <xs:attribute name="defRef" type="xs:positiveInteger" use="optional"/> </xs:complexType>

References

- [*1] S. Appel, P. Kleber, S. Frischbier, T. Freudenreich, and A. Buchmann, “Modeling and execution of event stream processing in business processes,” Inf. Syst., vol. 46, pp. 140–156, 2014.
- [*2] J. Dehnert, J. Dehnert, P. Rittgen, and P. Rittgen, “Relaxed soundness of business processes,” Lect. notes Comput. Sci., pp. 157–170, 2001.
- [*3] M. Fellmann, S. Bittmann, A. Karhof, C. Stolze, and O. Thomas, “Do We Need a Standard for EPC Modelling? The State of Syntactic, Semantic and Pragmatic Quality,” in 5th International Workshop on Enterprise Modelling and Information Systems Architectures (EMISA), 2013, pp. 103–116.
- [*4] F. Gottschalk, M. Rosemann, and W. M. P. van der Aalst, “My Own Process: Providing Dedicated Views on EPCs,” 4th GI Work. “EPC 2005 – Bus. Process Manag. with EPCs,” vol. 167, no. December, pp. 156–175, 2005.
- [*5] A. W. Scheer, O. Thomas, and O. Adam, Process Modeling using Event-Driven Process Chains. 2005.
- [*6] J. Mendling, G. Neumann, and M. Nüttgens, “Yet another event-driven process chain,” in Lecture Notes in Computer Science, 2005, vol. 3649, p. 428.
- [*7] F. Gottschalk, W. M. P. van der Aalst, and M. H. Jansen-Vullers, “Merging Event-Driven Process Chains,” pp. 418–426, 2008.
- [*8] O. Kopp, M. Wieland, and F. Leymann, “External and Internal Events in EPCs : e2EPCs,” 2nd Int. Work. event-driven Bus. Process Manag., pp. 381–392, 2010.
- [*9] A.-W. Scheer, M. Nüttgens, and V. Zimmermann, “Objektorientierte Ereignisgesteuerte

Prozeßkette (oEPK): Methode und Anwendung," Veröffentlichungen des Instituts für Wirtschaftsinformatik (IWi), Univ. des Saarlande, no. 141, 1997.

- [*10] B. List and B. Korherr, "A UML 2 Profile for Business Process Modelling *," vol. 205, pp. 85–96, 2005.
- [*11] Mendling: Event Driven Process Chains - Metrics for Process Models, Volume 6 of the series Lecture Notes in Business Information Processing, pp. 17-57, 2009.
- [*12] M. Nüttgens, F. J. Rump "Syntax und Semantik Ereignisgesteuerter Prozessketten", Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen - Promise 2002, P.69
- [*13] K. van Hee, O. Oanea, and N. Sidorova, "Colored Petri Nets to Verify Extended Event-Driven Process Chains", OTM Confederated International Conferences "On the Move to Meaningful Internet Systems", 2005, pp. 183-201.

Category:
Meta Model