



**Life-cycle of a program represented by a Generalized net**  
**Georgi Valchev**

Generalized Nets (see, e.g., [1]) are instrument for description of parallel flowing processes in the time. They implicate Petri Nets into.

One Generalized Net is formed by many transitions. Transitions are

combination of places, marked as “” and symbol “”. A transition where an arc enter is called enter position. A transition where an arc exit is called exit position. On every transition is put together index matrix of transition with measure  $m \times n$ . In this matrix  $m$  and  $n$  are number of enter positions and exit positions. On every two positions, enter and exit, corresponds predicate, which is a element form the matrix.

In this article I will represent life-cycle of a program with a generalized net. The life-cycle begin when we receive contract from a client for some application. Next step is analyst and design of the system according to client requirements. The process continue with development. In this stage, usually is possible to use many ready staff like functions, classes, libraries or even pieces from other projects, developed before. This is the previous experience. After development begin stage of fixing and changing the code (bug fixing) as a result of unsuccessfully tests. Here we can use previous experience in fixing bugs. One very important step is testing the program by Quality assurance team. They can create test scenarios by using client requirements. Also can use ready test cases and data given by the client. When program pass successfully all available tests, it is ready for “beta” version. If tests are not passed successfully the program is returned to stage of bug fixing. In “beta” stage the program is present to the client for a deep analysis. Also QAs can continue exploit the program with new cases. Ones, when all test are successfully passed and client is approved the product, is time for official release. This stage include feature support like adding new functional and /or possibly bugs fixing. The training of crew of client is not part of this article.

The Generalized Net that describe this process is present on fig.1.

In this article I describe the life-cycle of a software program. I will work without time components.

The transition  $z1$  is receiving contract from a client for some application. It is described like:

$$z1 = \langle \{l1\}, \{l2, l3, l4, l5\}, r1, v(l1) \rangle,$$

where:

$l1$  – become a token symbolized a client with some project,

$l2$  – the token pass to characteristic “description of the program, which client want”,

$l3$  – whit characteristic “provided from client test data”,

$l4$  – whit characteristic “additional test resources”. This token include more detailed data for test or crew member of client that can test in details and appraise the program,

$l5$  – whit characteristic “necessity of support for the client”. The token include remove defects in the program and adding new functional,

$r1$  is transition matrix:

	$l2$	$l3$	$l4$	$l5$
$l1$	true	true	true	true

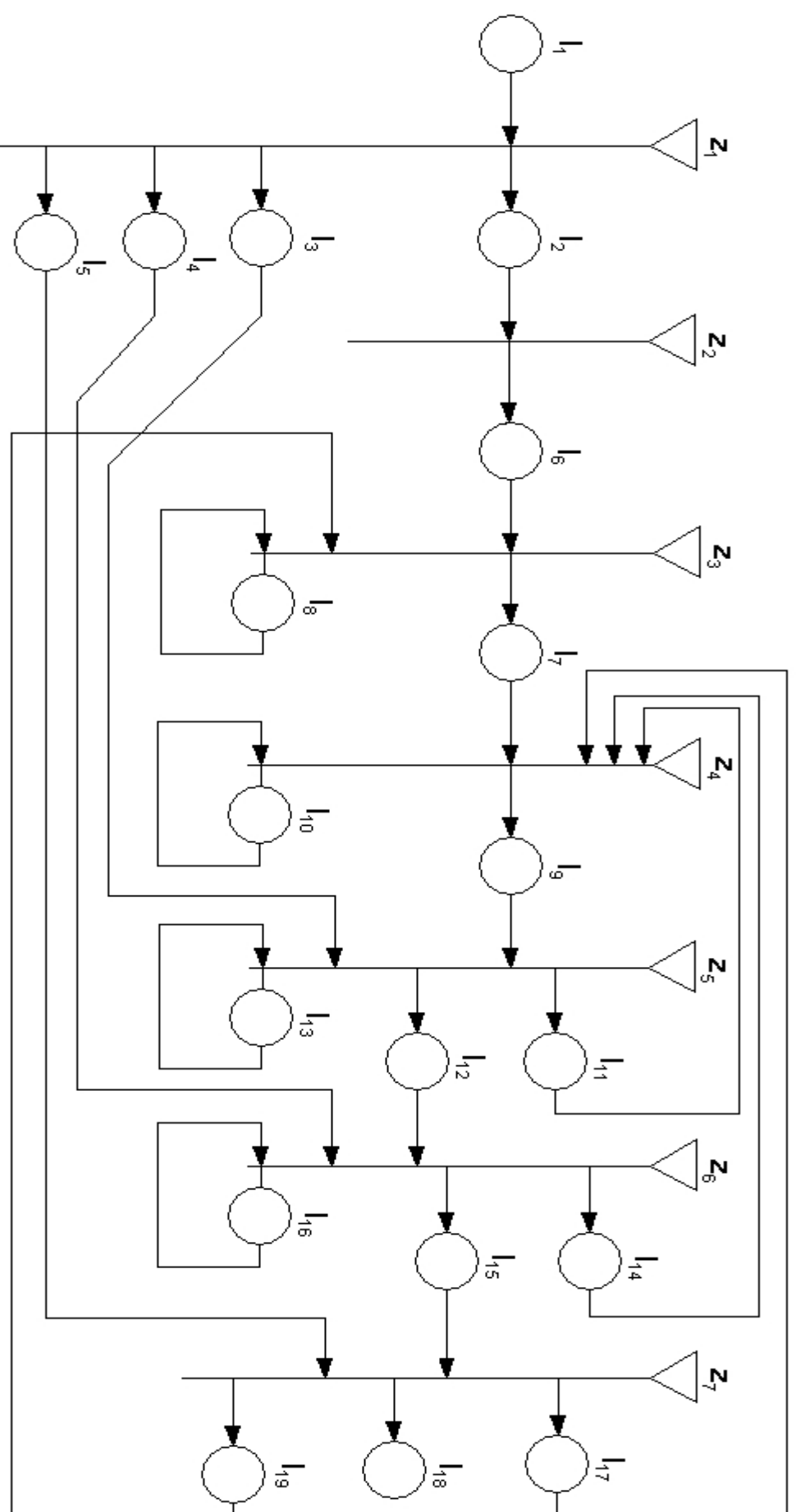


Fig. 1

The transition  $z_2$  is a design of the program. It is described like:

$$z_2 = \langle \{l_2\}, \{l_6\}, r_2, v(l_2) \rangle,$$

where:

$l_6$  – whit characteristic “decisions how to build the program”.

$r_2$  is transition matrix:

	$l_6$
$l_2$	true

In transition  $z_3$  begin development of the program. For this purpose is possible to use many ready staff like functions, classes, libraries or even pieces from other projects, developed before. The transition  $z_3$  is described like:

$$z_3 = \langle \{l_6, l_8, l_{19}\}, \{l_7, l_8\}, r_3, \wedge(l_8, v(l_6, l_{19})) \rangle,$$

where:

$l_7$  – whit characteristic “the program after development”,

$l_8$  – whit characteristic “previous experience”

$l_{19}$  – whit characteristic “experience after successful developed the program”,

$r_3$  is transition matrix:

	$l_7$	$l_8$
$l_6$	true	false
$l_8$	$w_{8,7}$	true
$l_{19}$	false	true

$w_{8,7}$  is a predicate:

$w_{8,7}$  = “in place  $l_6$  is received token what have analog in previous experience. The token from  $l_6$  is combined whit those part from previous experience and become in place  $l_7$ ”.

The transition  $z_4$  present removing defects in the program. Here also can be used previous experience in bug fixing process. It is described like:

$$z_4 = \langle \{l_7, l_{10}, l_{11}, l_{14}, l_{17}\}, \{l_9, l_{10}\}, r_4, \wedge(l_{10}, v(l_7, l_{11}, l_{14}, l_{17})) \rangle,$$

where:

$l_{10}$  – whit characteristic “previous experience in solving similar problems”,

$l_{11}$  – whit characteristic “rise a problem on test stage”,

$l_{14}$  – whit characteristic “rise a problem on beta stage or requirements for additional functional from client”,

$l_{17}$  – whit characteristic “rise a problem on final version or requirements for additional functional”,

$l_9$  – whit characteristic “the program after fixing some problem or adding new functional”,

$r_4$  is transition matrix:

	l9	l10
l7	true	false
l10	w10,9	true
l11	true	false
l14	true	false
l17	true	false

w10,9 is a predicate:

w10,9 = “in place l7 is received token what have analog in previous experience. The token from l7 is combined whit those part from previous experience and become in place l9”.

In transition z5 begin testing of the program from QAs. This is a internal testing form those how develop the product before release the beta version. Here also can be used previous experience in testing the program. It is described like:

$z5 = \langle \{l3, l9, l13\}, \{l11, l12, l13\}, r5, \wedge(l9, v(l3, l13)) \rangle$ ,

where:

l12 – whit characteristic ”successfully pass all available tests”. Now the program is ready for beta version,

l13 – whit characteristic “created previous tests for the program”,

r5 is transition matrix:

	l11	l12	l13
l3	w9,11	$\neg w9,11$	false
l9	w9,11	$\neg w9,11$	false
l13	w13,11	$\neg w13,11$	true

w3,11, w9,11 and w13,11 are following predicates:

w3,11 = w9,11 = “in place l9 is received token what fail to pass the tests”. This token proceed to place l11,

w13,11 = “in place l9 is received token what have analog in previous experience. The token from l9 is combined whit those part from previous experience. The test whit newly formed token fail”.

The transition z6 present rise to beta version of the program. Here become a test on these release and also can be used previous experience in testing the previous beta of the program. On this stage program is showed to the client and he can test it by his crew. It is described like:

$z6 = \langle \{l4, l12, l16\}, \{l14, l15, l16\}, r6, \wedge(l12, v(l4, l16)) \rangle$

where:

l15 – whit characteristic ”successfully pass all available tests”. Now the program is ready for official release version,

l16 – whit characteristic “created previous tests for beta test of the program”,

r6 is transition matrix:

	l14	l15	l16
l4	w12,14	$\neg$ w12,14	false
l12	w12,14	$\neg$ w12,14	false
l16	w16,14	$\neg$ w16,14	true

w4,14, w12,14 and w16,14 are following predicates:

w4,14 = w12,14 = “in place l12 is received token what fail to pass the tests”. This token proceed to place l14,

w16,14 = “in place l12 is received token what have analog in previous experience. The token from l9 is combined whit those part from previous experience. The test whit newly formed token fail”.

The transition z7 present rise to official version of the program. This is a version that client verify working correctly and it is ready to use in his work. It is described like:

$z7 = \langle \{l5, l15\}, \{l17, l18, l19\}, r7, \wedge(l5, l15) \rangle$

where:

l18 – whit characteristic ”everything is fine and client is satisfy”,

r7 is transition matrix:

	l17	l18	l19
l5	w5,17	$\neg$ w5,17	$\neg$ w5,17
l15	w5,17	$\neg$ w5,17	$\neg$ w5,17

w15,17 and w5,17 are following predicate:

w5,17 = w15,17 = “in place l15 is received token what raise to something different from normal work of the program.”. This token proceed to place l17.

## Reference

[1] Atanassov, K., Generalized nets, World Scientific, Singapore, New Jersey, London 1991.