CAPP CUSTOMIZATION ON THE BASE OF OBJECT-ORIENTED APPROACH

Alexander Sharmazanashvili
CAD/CAM Division
Georgian Technical University
sharm@gtu.edu.ge

ABSTRACT

Implementation of Group Technology in manufacturing processes planning brings necessity in CAPP customization. System customization often called third party developing, provides opportunity to create and built in existing software own modules and establish custom software. However, the development process of customer software is time consuming, difficult and requires highly skilled staff with good background in mathematics, programming and manufacturing technology. The paper describes an object-oriented approach of CAPP customization, simplifying customer software development process, which has been developed at Georgian Technical University.

KEYWORDS

CAPP, Group technology, typification, System architecture, Weight parameters, Object classes, Geometrical objects, Customer software.

1. PROBLEM

Group Technology has become an important technological innovation of Computer Aided Process Planning (CAPP). Several manufacturing companies have been widely applied this methodology in multi-product development (Jiang Wen Bing. et al., 1992) and small-batch production.

The core of Group Technology is typification. Typical decisions are made regarding to manufacturing processes being carried out for each particular part family (Mitrofanov S.P., 1959). Typification is made on the different levels of process description. Usually, these levels are including typical sequence of different operations, like turning, milling, drilling, etc. In other cases it will be typical structure of each operation with formal description of cutting instrument, fixtures, machining stocks, tool movement rules, cutting conditions, respective CNC subroutines, etc.

According to each level of typification, the corresponding decision-making models are built and realized as customer software of CAPP. On the base of this software, for each part from the given part family, process planning activity is carried out (Tsvetkov V.D., 1972). Therefore, while for the each part family typical decisions have to be separated with formation of corresponding customer software, causing the necessity of CAPP customization for each particular case.

The paper describes below a CAPP customization methodology for the level of manufacturing process description, including the machining stock, tool movement rules, cutting conditions and CNC subroutines. Usually, customer software development for given level of process typification is time consuming and difficult, while development of geometrical calculations algorithms in conjunction with the unique manufacturing experience formalization is required. The programming activity therefore requires a highly skilled CAPP user with a good background in mathematics, programming and manufacturing technology.

2. CONSIDERATION OF CAPP ARCHITECTURES

In CAD/CAM division at Georgian Technical University investigations are made in order to simplify CAPP programming activity (Sharmazanashvili A., 1998). Different CAPP architectures in this case where considered. For comparative analysis four weight parameters where separated, helping to measure software development difficultness:

- Mathematical difficultness
- Logical difficultness
- Manufacturing technology difficultness
- Basic computer skills.

2.1. Ordinary architecture

In ordinary architecture *two* main units can be separated – CAPP engine and customer software (Figure 1). CAPP engine provides basic functions of system - graphical interaction, feature recognition, interpretation, visualization and documentation.

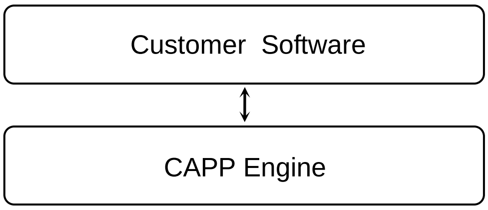


Figure 1. CAPP ordinary architecture

Customer software in this case provides system customization for each part family and contains the full algorithm of typical manufacturing process realization. Therefore its development process is most difficult and the above mentioned weight parameters will be measured as 100% of difficultness.

2.2. Object oriented architecture

Different values will be received in case of object oriented approach were implemented. While customer software includes both, geometrical calculation algorithms and technological algorithms, it's preferable to separate *two* main classes of objects: geometrical and technological (Sharmazanashvili A., 1997). The purpose of geometrical object is calculation of original shapes from typical shapes presented in the parametric form

$$\mathbf{D:\Gamma \rightarrow D'} \qquad (1)$$

D = parametrical description of typical shape

D' = original description of considered part shape in the form of vector; shape is represented by a sequence of support point and numerical values of each point coordinates

Γ = transformation rules.

Usually, various types of shapes are associated with the given part family. Therefore, the class of geometrical objects is divided on several subclasses of objects.

In the same way, technological objects are intended for calculation of tool path geometry and process condition parameters.

$$\mathbf{P:\Pi \rightarrow P'} \qquad (2)$$
$$\mathbf{C:\Omega \rightarrow C'} \qquad (3)$$

P,C = typical descriptions

P' = tool path geometry in the form of sequence of support points and its numerical coordinates in machine axis

C' = process condition parameters

Γ, Ω = transformation rules.

Different types of tool paths are usually separated and corresponding subclasses of objects are associated with them. Also, there are several process condition calculation rules with corresponding sub-classes (Sharmazanashvili A., et al., 1998).

The above mentioned object classes are unified as a common part of the customer software, called object system.

Another part of customer software is X-system, where the objects from different classes are linked in order to build process related algorithm and express typical manufacturing process originality. The main tasks of X-system programming are:

1) Selection of objects from object system
2) Provision of information linkage of objects through the set of assigns
3) Connection of X-system input parameters with parameters associated with objects.

The object-oriented CAPP architecture in this case contains the different configuration levels of customization (Figure 2).

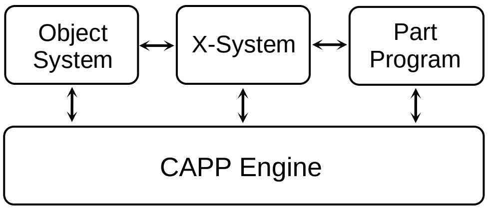


Figure 2. Object oriented architecture of CAPP

1st level unifies object system with CAPP engine and provides system general customization feature on the various types of manufacturing process – turning, milling, drilling, etc.

2nd level, unifies X-system with object system and CAPP engine and provides system customization feature on various part families.

3rd level, unifies part program with X-system, object system and CAPP engine and provides system customization feature on each part from part family.

2.3. Comparative analysis

Development of object system is characterized with the necessity of a strong background in mathematical formalization and geometrical transformation methods. Also, a deep knowledge of logic and programming methods of calculation steps, logical branches, cycles, parameters inspection procedures, etc. is required. Therefore the weight parameters value of mathematical and logical difficultness will be high and close to case of ordinary architecture. Manufacturing technology difficultness will be low, while object system does not express any typical manufacturing process and also a deep knowledge in manufacturing technology is not required. Special procedures of object programming, also increase requirements in basic computer skills.

Regarding to X-system, mathematical and logical difficultness will be low, because geometrical transformations are not required and only algorithms of object parameters definition will be worked out.

Manufacturing technology difficultness will be high. While X-system expresses the typical manufacturing process originality, a strong background in manufacturing technology and process formalization methods are needed. Basic computer skills will be average.

As a result, the generation of solution and corresponding program for concrete part is much simplified. Mathematical and logical contribution is very low; also low is process programming difficultness, while process description requires only the definition of objects from X-system.

A comparatively high level of basic computer skills is required to interact with X-system and CAPP engine.

Figure 3, below represents the results of the above mentioned analysis. As it is shown, the summarized difficultness of customer software development is cutting down from object system to part program.

The advantage of the suggested object oriented architecture is that the most difficult part of the customer software, that is the object system, has to be worked out only once, when the CAPP general customization on the manufacturing process types is made. While in ordinary architecture this is necessary to be done every time when manufacturing process typification is required.

Description	*Skills*	*Background*	*Customization*
Object system	High	• Mathematics • Programming	Manufacturing process type
X-system	Med.	• Technology	Family of parts
Part program	Low	• Operator	Part

Table 1.

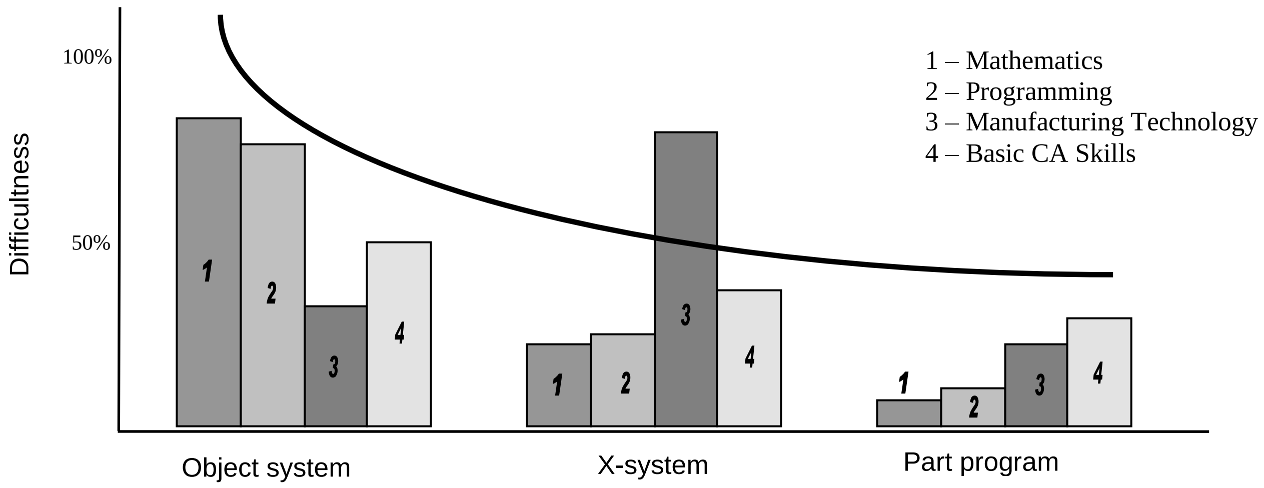


Figure 3. Comperative analysis of customer software development difficultness in object-oriented architecture

3. DESCRIPTION OF OBJECT SYSTEM

An object system for turning and X-system for FLANGE part family, was built in CAD/CAM division at Georgian Technical University on the base of suggested approach. 10 main geometrical classes, semi-open cylindrical, open cylindrical, closed cylindrical, open face, closed face, semi-open grooving, open grooving, closed grooving, cylindrical conjunction and grooving conjunction were separated. Also, 28 tool movement and 5 process conditions optimization objects were built. 142 objects of geometrical transformations were worked out for each given classes of object. They permit to describe up to 95% of turning part surfaces. In special cases, additional objects have to be worked out.

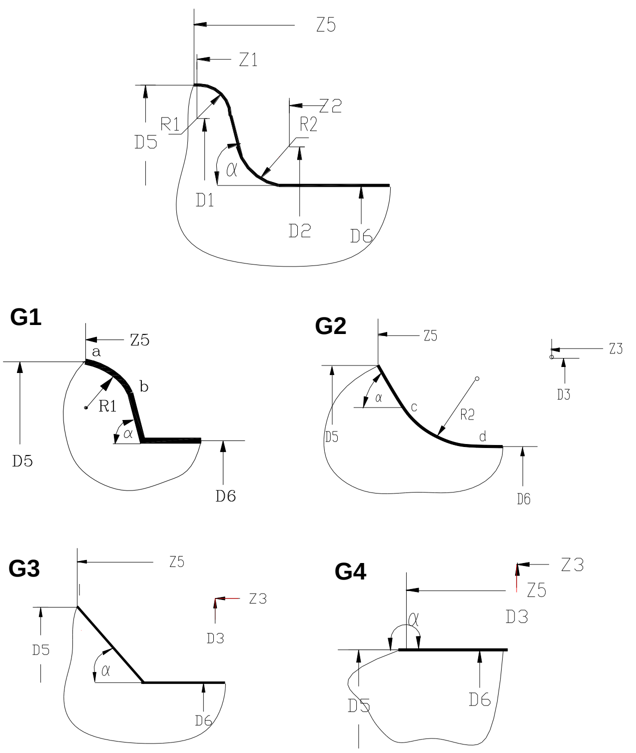
3.1. Geometrical Objects

Three main features characterize each class of geometrical object:

- Structure, describes a set of shapes the object geometry consists of
- Topology, describes how typical shapes in object geometry are connected
- Parameterization, describes array of parameters, necessary for object geometry formal representation.

Figure 4, describes semi-open cylindrical class of objects.

Class : SEMI_OPEN_CYLINDRICAL_STAIR
Objects : {G1, G2, G3, G4, G5, G6, G7}



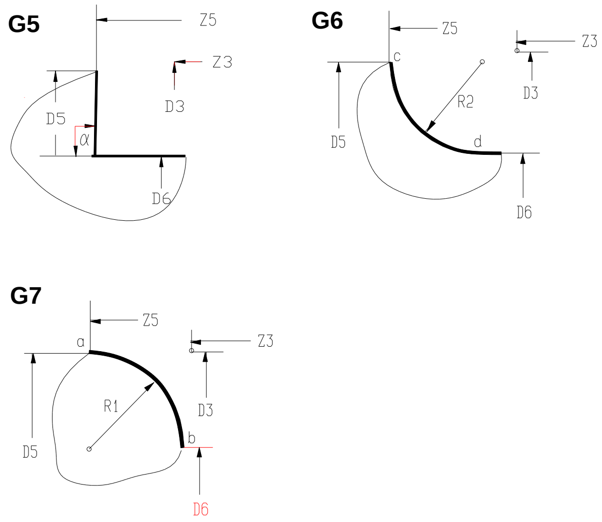


Figure 4. Semi-open cylindrical class of objects

This class is intended for typical shapes formed by tangential conjunction of two arcs with a conical line and has topology “Arc-Line-Arc-Line”. The parameterization describes 12 geometrical parameters:

Class *‘Arc-Line-Arc-Line’*:
{D5,Z5,D6,D3,Z3,ALF,R1,R2,D1,Z1,D2,Z2}
(4)

The original set of values of parameters are corresponding to each object from the considered class. Among this set, in each case, the key parameters can be identified, which describes condition of the object separation. For example zero value of parameter *R2* describes condition for separation of G1 object.

Class	Parameters		
	Required	Key	Additional
G1:’Arc-Line-Line’	D5,Z5,D6, ALF,D3,Z3, R1	R2=0	D1,Z1,D2, Z2

Table 2

If *R1* equals to zero, another object G2 ‘Line-Arc-Line’ is received

Class	Parameters		
	Required	Key	Additional
G2:’Line-Arc-Line’	D5,Z5,D6, ALF,D3,Z3, R2	R1=0	D1,Z1,D2, Z2

Table 3

In case, if both *R1* and *R2* parameters equal to zero, according to value of *ALF* parameter, the following objects are separated

Class	Parameters		
	Required	Key	Additional
G3:’Line-Line’	D5,Z5,D6, D3,Z3	R1=0 R2=0 0<ALF<90	D1,Z1,D2, Z2
G4:’Line’	D5,Z5,D6, D3,Z3	R1=0 R2=0 D5=D6 ALF=180	D1,Z1,D2, Z2
G5:’Line-Line’	D5,Z5,D6, D3,Z3	R1=0 R2=0 ALF=90	D1,Z1,D2, Z2

Table 4

If *ALF* equal to zero, whether *R1* equal to zero, or *R2* equal to zero, defines conditions for separation of G6 or G7 objects.

Class	Parameters		
	Required	Key	Additional
G6:’Arc’	D5,Z5,D6, D3,R2	R1=0 ALF=0	D2, Z2, Z3
G7:’Arc’	D5,Z5,D6, D3,R1	R2=0 ALF=0	D1, Z1, Z3

Table 5

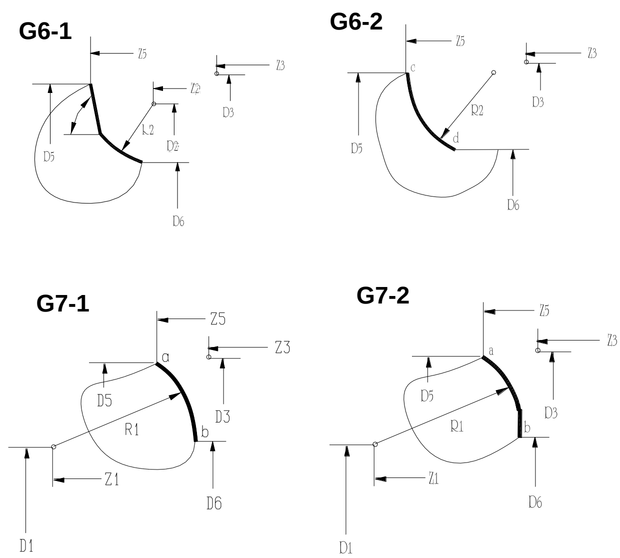
Values of additional parameters identify conditions for separation of sub-classes from the above considered class of objects.

In case, when *D1,Z1,D2,Z2* parameters values are not equal to zero, separation of sub-objects with a topology of non-tangential conjunctions of arc and line is carried out. Table 6 describes all sub-objects to be received.

Class	Sub-class	Parameters		
		Required	Key	Additional
G6	G6-1:’Arc’	D5,Z5,D6, D3,Z3,R2, Z2,D2	R1=0 0<ALF ≤90	D1,Z1
G6	G6-2:’Line-Arc’	D5,Z5,D6, D3,Z3,R2, Z2,D2	R1=0 ALF=0	D1,Z1
G7	G7-1:’Arc’	D5,Z5,D6, D3,Z3,R1, Z1,D1	R2=0 ALF=0	D2,Z2
G7	G7-2:’Arc-Line’	D5,Z5,D6, D3,Z3,R1, Z1,D1	R2=0 0<ALF ≤90	D2,Z2
G1	G1-1:’Arc-Line-Line’	D5,Z5,D6, D3,Z3,R1, ALF	R1=0 D1=0 Z1=0 Z2≠0	D2
G2	G2-1:’Line-Arc-Line’	D5,Z5,D6, D3,Z3,R2, ALF	R1=0 D2=0 Z2≠0	D1,Z1
G0	G0-1:’Arc-Line-Arc-Line’	D5,Z5,D6, D3,Z3,R2, R1,ALF	Z1=0 D1=0 D2=0 Z2≠0	-

Table 6

Particular attention in this case is paid to *Z2* parameter, while non zero value of *Z2* with different combination of non zero values of other parameters – *Z1,ALF,R1,R2* identify conditions for separation sub-objects with non-tangential conjunctions of arc with the internal line. In rest of cases non-tangential conjunctions of arc are made with external lines (lines from other objects). Corresponding shapes are presented on figure 5.



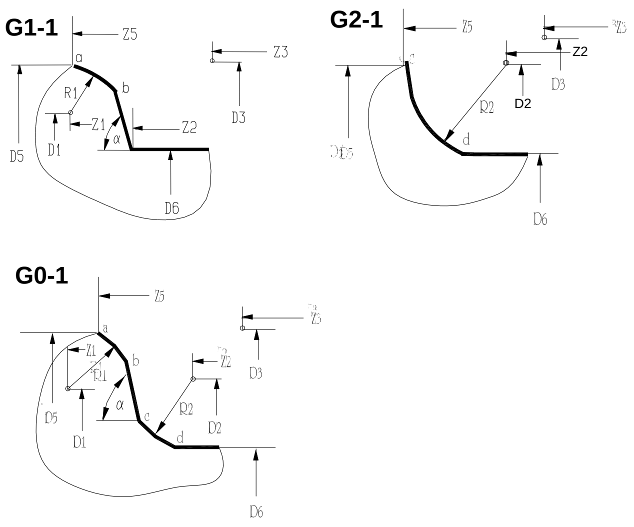


Figure 5. Object sub-classes

Mathematical model of each object contains a set of equations, which express functional dependence of shapes support points on object parameters. According to parameterization, each support point is described by different set of parameters and for its calculation corresponding equation have to be separated.

The considered class of semi-open cylindrical objects contains five support points T1-T2-T3-T4-T5 (Figure 6).

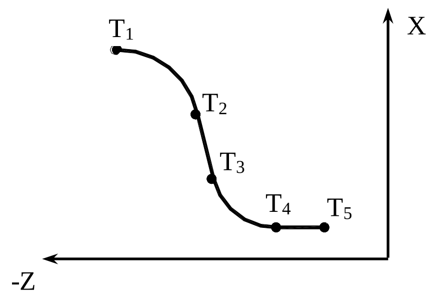


Figure 6. Geometrical structure of semi-open cylindrical objects

T_1 is described directly by parameters {D5,Z5} from (4)

$$\begin{Bmatrix} x_1 = \dfrac{D5}{2} \\ z_1 = Z5 \end{Bmatrix} \qquad (5)$$

T_2 is described by {D5, Z5, R1, α}

$$\begin{Bmatrix} x_2 = \dfrac{D5}{2} - R1 \cdot (1 - \cos\alpha) \\ z_2 = Z5 - R1 \cdot \sin\alpha \end{Bmatrix} \qquad (6)$$

T_3 is described by {D5, Z5, D6, R1, R2, α}

$$\begin{Bmatrix} x_3 = \dfrac{D6}{2} - R2 \cdot (1 - \cos\alpha) \\ z_3 = Z5 - R1 \cdot \sin\alpha + ctg\alpha \cdot [\dfrac{D5 - D6}{2} - \\ - (R1 + R2) \cdot (1 - \cos\alpha)] \end{Bmatrix}$$

(7)

T_4 is described by {D5, Z5, D6, R1, R2, α}

$$\begin{Bmatrix} x_4 = \dfrac{D6}{2} \\ z_4 = Z5 - \sin\alpha \cdot (R1 - R2) + ctg\alpha \cdot [\dfrac{D5 - D6}{2} - \\ - (R1 + R2) \cdot (1 - \cos\alpha)] \end{Bmatrix}$$

(8)

T_5 is described by {D6, Z3}

$$\begin{Bmatrix} x_5 = \dfrac{D6}{2} \\ z_5 = Z3 \end{Bmatrix} \qquad (9)$$

In case when the D1, Z1, D2, Z2 is not equal to zero, non-tangential conjunctions of arc and line in T_1 and T_4 are carried out. So, in (6), (7), (8), D5 replaced with D1 and D6 with D2.

The original set of equations have to be received from described general set for each object.

$\boxed{\text{G01}}$ According to table 2, T_3=T_4. Therefore,

$$\begin{Bmatrix} x_1 = \dfrac{D5}{2} \\ x_2 = \dfrac{D5}{2} - R1 \cdot (1 - \cos\alpha) \\ x_3 = x_4 = x_5 = \dfrac{D6}{2} \\ z_1 = Z5 \\ z_2 = Z5 - R1 \cdot \sin\alpha \\ z_3 = z_4 = Z5 - R1 \cdot \sin\alpha + ctg\alpha \cdot [\dfrac{D5 - D6}{2} - \\ - R1 \cdot (1 - \cos\alpha)] \\ z_5 = Z3 \end{Bmatrix}$$

(10)

$\boxed{\text{G02}}$ According to table 3, T_1=T_2. Therefore,

$$\begin{cases} x_1 = x_2 = \dfrac{D5}{2} \\ x_3 = \dfrac{D6}{2} + R2 \cdot (1-\cos\alpha) \\ x_4 = x_5 = \dfrac{D6}{2} \\ z_1 = z_2 = Z5 \\ z_3 = Z5 + ctg\alpha \cdot [\dfrac{D5-D6}{2} - R2 \cdot (1-\cos\alpha)] \\ z_4 = Z5 - R2 \cdot \sin\alpha + ctg\alpha \cdot [\dfrac{D5-D6}{2} - R2 \cdot (1-\cos\alpha)] \\ z_5 = Z3 \end{cases}$$

(11)

$\boxed{G03}$ According to table 4, T_1=T_2 and T_3=T_4. Therefore,

$$\begin{cases} x_1 = x_2 = \dfrac{D5}{2} \\ x_3 = x_4 = x_5 = \dfrac{D6}{2} \\ z_1 = z_2 = Z5 \\ z_3 = z_4 = Z5 - ctg\alpha \cdot (\dfrac{D5 - D6}{2}) \\ z_5 = Z3 \end{cases} \quad (12)$$

$\boxed{G04}$ According to table 4, T_1=T_2 and T_3=T_4=T_5. Therefore,

$$\begin{cases} x_1 = x_2 = x_3 = x_4 = x_5 = \dfrac{D5}{2} \\ z_1 = z_2 = Z5 \\ z_3 = z_4 = z_5 = Z3 \end{cases} \quad (13)$$

$\boxed{G05}$ According to table 4, T_1=T_2 and T_3=T_4. Therefore,

$$\begin{cases} x_1 = x_2 = \dfrac{D5}{2} \\ x_3 = x_4 = x_5 = \dfrac{D6}{2} \\ z_1 = z_2 = z_3 = z_4 = Z5 \\ z_5 = Z3 \end{cases} \quad (14)$$

$\boxed{G06}$ According to table 5, T_1=T_2=T_3 and T_4=T_5. Therefore,

$$\begin{cases} x_1 = x_2 = x_3 = \dfrac{D5}{2} \\ x_4 = x_5 = \dfrac{D6}{2} \\ z_1 = z_2 = z_3 = Z5 \\ z_4 = z_5 = Z5 - R2 \end{cases} \quad (15)$$

$\boxed{G07}$ According to table 5, T_2=T_3=T_4=T_5. Therefore,

$$\begin{cases} x_1 = \dfrac{D5}{2} \\ x_2 = x_3 = x_4 = x_5 = \dfrac{D6}{2} \\ z_1 = Z5 \\ z_2 = z_3 = z_4 = z_5 = Z5 - R1 \end{cases}$$

(16)

For Sub-classis of objects, according to table 6, following models are separated:

$\boxed{G6\text{-}1}$ T_1=T_2 and T_4=T_5

$$\begin{cases} x_1 = x_2 = \dfrac{D5}{2} \\ x_3 = \dfrac{D2}{2} - R2 \cdot \cos\alpha \\ x_4 = x_5 = \dfrac{D6}{2} \\ z_1 = z_2 = Z5 \\ z_3 = Z5 - ctg\alpha \cdot (\dfrac{D5 - D2}{2} + R2 \cdot \cos\alpha) \\ z_4 = z_5 = Z3 \end{cases}$$

(17)

$\boxed{G6\text{-}2}$ T_1=T_2=T_3 and T_4 =T_5

$$\begin{cases} x_1 = x_2 = x_3 = \dfrac{D5}{2} \\ x_4 = x_5 = \dfrac{D6}{2} \\ z_1 = z_2 = z_3 = Z5 \\ z_4 = z_5 = Z3 \end{cases}$$

(18)

$\boxed{G7\text{-}1}$ T_2=T_3=T_4=T_5

$$\left\{\begin{array}{l} x_1 = \dfrac{D5}{2} \\ x_2 = x_3 = x_4 = x_5 = \dfrac{D6}{2} \\ z_1 = Z5 \\ z_2 = z_3 = z_4 = z_5 = Z3 \end{array}\right\}$$

(19)

$\boxed{\text{G7-2}}$ $T_3=T_4=T_5$

$$\left\{\begin{array}{l} x_1 = \dfrac{D5}{2} \\ x_2 = \dfrac{D1}{2} + R1\cdot\cos\alpha \\ x_3 = x_4 = x_5 = \dfrac{D6}{2} \\ z_1 = Z5 \\ z_2 = Z1 - R1\cdot\sin\alpha \\ z_3 = z_4 = z_5 = Z3 \end{array}\right\}$$

(20)

$\boxed{\text{G1-1}}$ $T_3=T_4$

$$\left\{\begin{array}{l} x_1 = \dfrac{D5}{2} \\ x_2 = \dfrac{D6}{2} + \dfrac{Z2\cdot(\dfrac{D1-D6}{2} + R1\cdot\cos\alpha)}{Z1 - R1\cdot\cos\alpha - ctg\alpha\cdot(\dfrac{D1-D6}{2} + R1\cdot\cos\alpha)} \\ x_3 = x_4 = x_5 = \dfrac{D6}{2} \\ z_1 = Z5 \\ z_2 = Z2 + \dfrac{Z2\cdot ctg\alpha\cdot(\dfrac{D1-D6}{2} + R1\cdot\cos\alpha)}{Z1 - R1\cdot\sin\alpha - ctg\alpha\cdot(\dfrac{D1-D6}{2} + R1\cdot\cos\alpha)} \\ z_3 = z_4 = Z2 \\ z_5 = Z3 \end{array}\right\}$$

(21)

$\boxed{\text{G0-1}}$ $T_1=T_2$

$$\left\{\begin{array}{l} x_1 = \dfrac{D5}{2} \\ x_2 = \dfrac{D1}{2} + R1\cdot\cos\alpha \\ x_3 = \dfrac{D2}{2} - R2\cdot\cos\alpha \\ x_4 = x_5 = \dfrac{D6}{2} \\ z_1 = Z5 \\ z_2 = Z1 - R1\cdot\sin\alpha \\ z_3 = Z1 - R1\cdot\sin\alpha + ctg\alpha\cdot[\dfrac{D1-D2}{2} - (R1 + \\ + R2)\cdot(1-\cos\alpha] \\ z_4 = Z2 + \sqrt{R_2^2 - (\dfrac{D2-D6}{2})^2} \\ z_5 = Z3 \end{array}\right\}$$

(22)

$\boxed{\text{G2-1}}$ $T_1=T_2$

$$\left\{\begin{array}{l} x_1 = x_2 = \dfrac{D5}{2} \\ x_3 = \dfrac{D6}{2} + R2\cdot(1-\cos\alpha) \\ x_4 = x_5 = \dfrac{D6}{2} \\ z_1 = z_2 = Z5 \\ z_3 = Z5 - ctg\alpha\cdot[\dfrac{D5-D6}{2} - R2\cdot(1-\cos\alpha] \\ z_4 = Z2 + \sqrt{R_2^2 - (\dfrac{D2-D6}{2})^2} \\ z_5 = Z3 \end{array}\right\}$$

(23)

Same objects and corresponding models are also built for the other classes.

3.2. Tool Movement Objects

Tool movement objects permit to describe the support points of tool path according to part shape separated by geometrical objects and current position of tool.

Three main classes of tool movement objects were separated for considered Object system:

M1 – 4 point closed cycle movement

M2 – 3 point closed cycle movement

M3 – Equidistant movement.

M1 objects provide calculation of P_2 conducted point from the P_1 starting point across the X or Z axis parallel line; also, P_3 and P_4 points are calculated to provide tool back movement. So, each object of the given class is characterized by 3 typical movements. Depending on weather, this movements are carried out fast, or on feederate, *two* different sub-classes of M1 objects are separated, with topology:

M1-1 –“Feederate->Fast->Fast”

M1-2 – “Feederate->Feederate->Fast”

M1-1 class of objects provide tool movement on feederate from P_1 starting point up to P_2 conjunct point, which is placed on part surface (Figure 7);

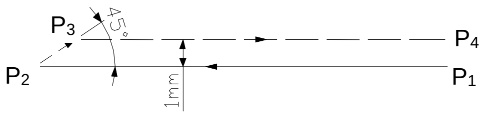


Figure 7. M1-1 class of tool movement object

then fast movement across the 45^0 angled line up to P_3 point with transferring on 1mm and back fast movement in P_4 point. 8 different objects are separated from given sub-class according to left-right,-up-down directions of P_1, P_2 movement and P_2, P_3 fast movement.

M1-2 class of objects provide tool movement on feederate from P_1 starting point to P_2 point (Figure 8); then feederate movement up to P_3 point across the part surface with transferring on predefined depth of cut (*t*) and back fast movement in P_4 point is carried out.

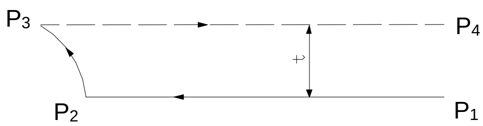


Figure 8. M1-2 class of tool movement object

Also, 8 different objects are separated from M1-2 class, according to left-right-up-down directions of P_1, P_2 and P_2, P_3 feederate movement.

M2 objects provide calculation of P_2 conjuncted point from the P_1 starting point across the X or Z axis parallel line. Tool movement is starting on feederate from P_1 point up to P_2 point and finished by back fast movement in P_1 point (Figure 9). So, all objects from given class have the same topology – “Fast-Fast”. According to left-right-up-down directions of P_1, P_2 movement, 4 different objects were separated.

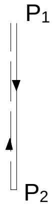


Figure 9. M2 class of tool movement object

M3 objects provide calculation of points for equidistant movement. Number of points is depends on the part shape, being “copied” during equidistant movement and is limited by geometrical objects. There are two sub-classes of M3 objects:

M3-1 provide equidistant movement with scaling (Figure 10) and M3-2, equidistant movement without scaling (Figure 11). According to left-right-up-down directions of movement, 4 different objects from each sub-class were separated.

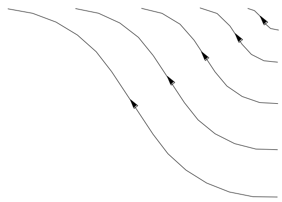


Figure 10. M3-1 class of tool movement object

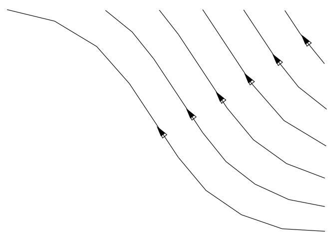


Figure 11. M3-2 class of tool movement object

Finally, 28 different tool movement objects were built.

3.3. Process Optimization Objects

Objects for optimization of process condition parameters were worked out from general representation of optimization model (Sharmazanashvili at Al., 1988). According to this representation optimal values of process parameters can be found by simultaneous solving of *two* equations of boundary conditions

$$\begin{cases} H = C_H \cdot V^{\alpha_H} \cdot S^{\beta_H} \cdot t^{\gamma_H} \\ \Phi = C_\Phi \cdot V^{\alpha_\Phi} \cdot S^{\beta_\Phi} \cdot t^{\gamma_\Phi} \end{cases} \qquad (24)$$

Gornev V.F. (1980) suggests to joint in the [H] class, equations that are relatively substantially depended on feederate (S) and in the [Φ] class, equations that are rather depended on cutting speed (V). As a result of analysis of boundary conditions described in above mentioned source, *five* different objects of optimization were separated:

[PV] – by restrictions of *P=Const* and *V=Const*, where, *P* - cutting force and *V* - cutting speed.

[ST] – by restrictions of *S=Const* and *T=Const*, where, *S* - feederate and *T* - tool life period.

[SN] – by restrictions of *S=Const* and *N=Const*, where, *N* – cutting power.

[MN] – by restrictions of *M=Const* and *N=Const*, where, *M* – cutting moment.

[MT] – by restrictions of *M=Const* and *T=Const*.

3.4. Programming Conditions

An object system software was built on the APT similar language for object-oriented engine Turbo T. System is placed into the text files where source code is presented. Turbo T engine carries out the interpretation of the source code and the generation of CLDATA. Language possibilities are described in Table 7.

Formal Parameters	A, B, C, A1, Z5, DEPTH, ...
Functions	SIN, COS, TAN, ATAN, ASIN, SQRT, ROUND, SQR
Mathematical expressions	A=SQRT(B) + SIN (ROUND (D*2/B)) / (C-1)^2
Labels	<Numbers>, <Strings>
Tool Axis motion operator	X <value>, Z <value>
Conditional movement operator	IF (<value>) <value>
Unconditional movement operator	Goto <value>
Cycle and assignment operators	Macro (<value>)
Cutting conditions defining operators	Federate (<value>) / Fast
Tool fix operator	Tool (<value>)

Table 7.

CONCLUSIONS

(a) Separation of *two* main parts in customer software brings the advantage of the suggested object oriented architecture of CAPP. While it brings possibility to move the most difficult part of customer software, related with geometrical transformations, to the lowest level of system architecture and carries out its development process only once during system customization.

(b) Manufacturing process-related algorithms are worked out on a higher level of system architecture. Corresponding customer software development process is carried out each time when process typification for group technology is necessary and it requires users with a strong knowledge of manufacturing technology only, without any strong background in mathematics and programming.

(c) Development of object system for turning at Georgian Technical University, showed that the corresponding tasks were well formalized and a language with pure programming ability can be implemented.

(d) For turning objects 142 geometrical, 28 tool movement and 5 optimization objects were built. They can easily share to different part families and permits to cover up to 95% of cases.

(e) Suggested approach can be adopted on other types of machining operations without any considerable changes.

REFERENCES

Jiang Wen Bing, Wang Zhi Bo, Sun Hou Fang (1992) "Group Technology Application in

China"; *in Technovation*, vol:12, Iss:8, pp.509-514

Mitrovanov S.P. (1959) "The Fundamentals of Group Technology"; *Leningrad*, p.265 (in Russian)

Tsvetkov V.D. (1972) "Implementation of Computers for Designing of Technological Process of Machining"; *Mashinostroenie,* Moscow, p.237 (in Russian)

Gornev V.F., (1989), "Integration of Technological operation control systems"; *Proceedings of first all union meeting of manufacturers*, Moscow, pp.41-43. (in Russian)

Sharmazanashvili A. (1998) "The State-of-the-art in trends of CAD/CAM Development"; *in Transactions of Georgian Technical University,* no.1 (417), pp.100-110

Sharmazanashvili A., Megrelishvili L (1998) "Feature-based approach in CAD/CAM/CNC Integration"; *in Transactions of 2nd International Symposium of Tools and Methods of Concurrent Engineering, 1998, 19-21 April, Manchester, UK*, pp.423-431

Sharmazanashvili A., Kharadze I. (1998) "Implementation of Heuristic Models for the Parametric Optimization of Turning"; *Proceedings of the ParametricCAD'98 workshop*, no.3, GTU,pp.54-62

Sharmazanashvili A. (1997) "Consideration of Para/CAM – the Object Oriented Process Planning Software"; *Proceedings of the ParametricCAD'97 workshop,* no.2, GTU, pp.7-15