

Top-down Paradigm in Engineering Software Integration

July 23, 2009

Petr R. Ivankov*

e-mail: * monstr3d@korolev-net.ru,

Abstract

The top-down approach of engineering software integration is considered in this paper. A set of advantages of this approach are presented, by examples. All examples are supplied by open source code.

1 Introduction

Historically engineering software integration is rather chaotic than planned. Different CAD products had been integrated with CAE and CAM products, Matlab had been integrated with Simulink etc. This integration history could be schematically presented in figure 1.

However chaotic way is not optimal way. For example C++ programming language had been designed chaotically. Languages of next generation (Java, C#) had been designed by planned way. C++ does not support automatic garbage collection. But any present day project should have it. So big C++ projects contain smart pointers those provide automatic garbage collection. Since smart pointer is not intrinsic C++ feature there exists a lot of versions of smart pointers. Integration of set of projects with different versions of smart pointers has a set of disadvantages. In this case integrated product is not clear, volume of its code is overloaded etc. Present day programming languages (Java, C#) support Reflection but C++ does not support it. However Reflection is very useful

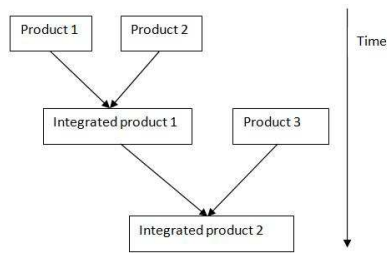


Figure 1: Chaotic integration of software products

feature for any big project. So big C++ projects have different versions of Reflection emulation. Comparing of intrinsic reflection and reflection emulation is presented in figure 2.

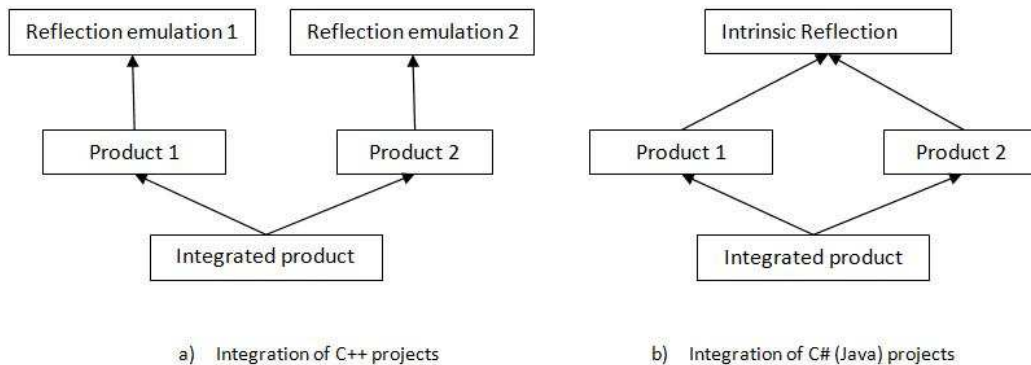


Figure 2: Reflection emulation and intrinsic reflection

In case of emulation we have additional code without essentially new functionality. If we would like integrate a set of projects by such way then we will obtain a lot of lumber. Such disadvantage have chaotic way of integration. But chaotic way has top-down design alternative. The top-down paradigm is presented in figure 3.

Let us explain meaning of figure 3. First of all we have pure abstract library devoted to science and engineering. This library is used by derived libraries devoted to different brunches of science and engineering. Libraries can be abstract. Integration can use other engineering software as third party. Libraries are being used by integrated products. Green, pig and purple color are used for libraries, third party and integrated products respectively. Italic font is used for abstract libraries.

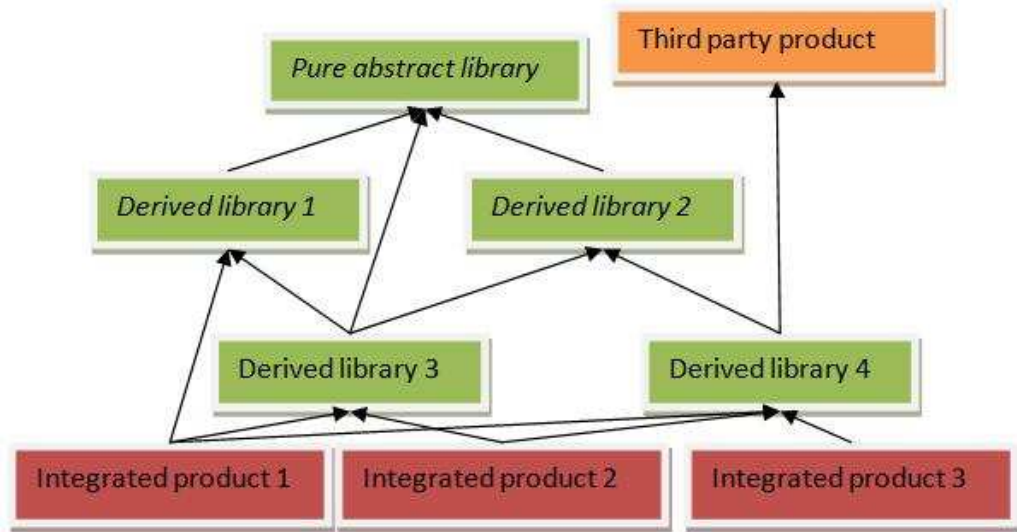


Figure 3: Top-down integration of engineering software products

2 Category theory as prototype

Pure abstract library for science and engineering should have very high level of abstraction. However this level of abstraction is already developed in math. In mathematics, category theory deals in an abstract way with mathematical structures and relationships between them: it abstracts from sets and functions to objects linked in diagrams by morphisms or arrows. High level of abstraction of Category Theory is explained in [1]. This book contains figure 4 with following text.



Figure 4: Object and arrow

We did not actually say what a and f are. The point is that they can be anything you like. a might be a set with f its identity function. But f might be a number, or a pair of numbers, or a banana, or the Eiffel tower, or even Richard Nixon. Likewise for a . Let us consider application of this abstraction. Database diagram and control systems diagram are presented in figure 5. Common features of these diagram are objects and arrows as abstract objects.

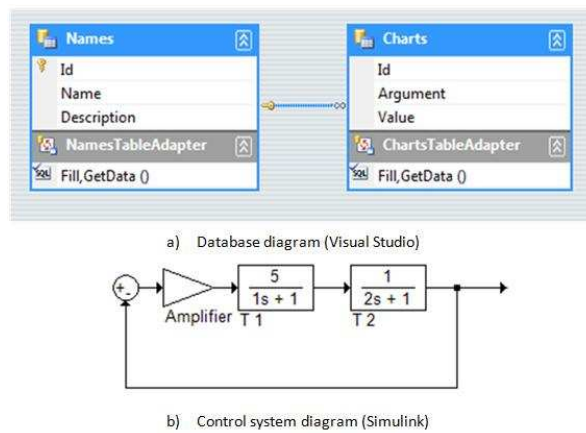


Figure 5: Database and control system diagrams

Figure 6 presents database and control systems diagram in single framework (There is no one to one map).

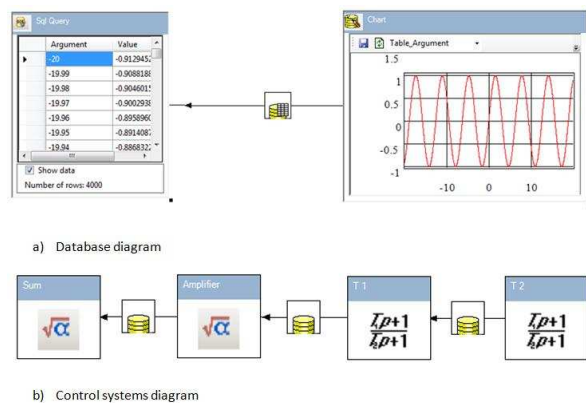


Figure 6: Database and control system diagrams in single framework

Now we would like to integrate both diagrams. Integrated diagram is presented in figure 7 Presented on 7 diagram has following meaning. Database contains recorded input signals of control systems. This diagram enable us to define response of control system.

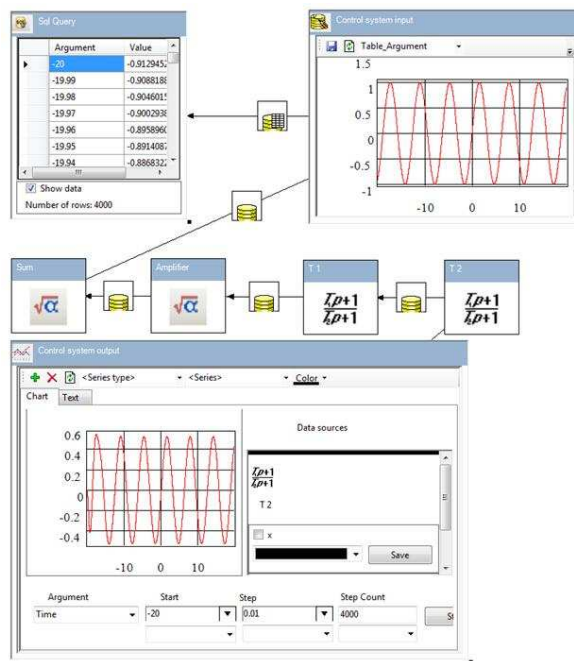


Figure 7: Integrated database and control system diagram

3 Present day state

Explanation of idea usefulness is difficult without any implementation. This idea shall find further development. However a lot of features are already implemented. These features are presented in figure 8. Presented in figure 8

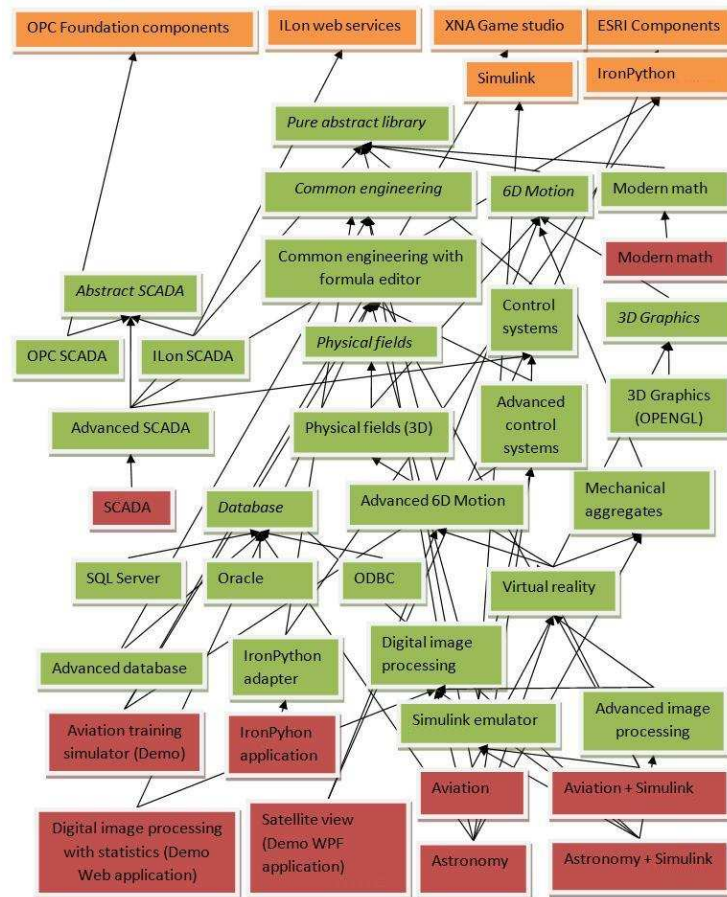


Figure 8: Hierarchy of assemblies

scheme reflects only part of implemented features. Source code can be downloaded from <http://www.mathframe.com>. It is worth to note whole product is not a sum of presented in figure 8 features only. These features can interact and interaction make product much more powerful. This thesis will be shown below by examples. This hierarchy can inspire some questions. One of reasonable question is: "Why we need abstract physical field library?" Now this software supports physical fields in 3D space. However this software is declared as universal and should support 2D physical fields in future. And 2D fields library shall

be inherited from abstract physical field library. Universality had been provided from the very beginning of development. Demo applications show prospects of further development.

4 Elementary examples

Examples of this section are not not related to real science and engineering problems. These examples are rather textbook which is very facile for idea explanation.

4.1 Physical fields

One physical phenomenon acts to another one and there is backward dependence. This thesis can be exhibited by example presented in figure 9. We have

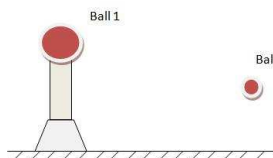


Figure 9: Charged balls

two charged balls **Ball 1** and **Ball 2**. Electrostatic force which act to **Ball 2** depends on relative position **Ball 2** with respect to **Ball 1**. Otherwise the relative position as time function depend on motion of **Ball 2** and therefore relative position depends on electrostatic force. Common consideration of 6D motion and physical fields make software much more effective. Now we would like simulate motion of **Ball 2**. Simulation scheme is presented in figure 10. Left

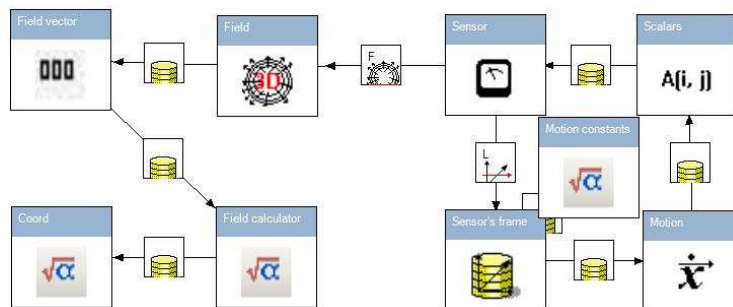


Figure 10: Charged balls motion simulation

part of this picture represents electrostatic field of **Ball 1**. Objects **Coord**,

Field calculator and **Field vector** provide calculations of next formula of electrostatic intensity:

$$E = \frac{kr}{|r|^3};$$

where r is 3D vector of relative position with respect to charge, k is coefficient which depends on charge of **Ball 1** and system of units. This calculation is separated for performance issue. Object **Coord** calculates $\frac{k}{|r|^3}$, object **Field calculator** calculates components of 3D vector $\frac{kr}{|r|^3}$ and **Field vector** assembles these components into vector. All these calculation are used by **Field** object. Properties of the **Field** object are presented on figure 11.

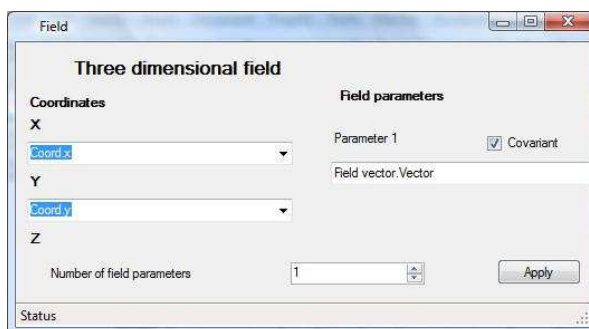


Figure 11: Properties of **Field** object

These properties have following meaning. First of all field parameter is output vector of **Field vector**. Figure 12 explains meaning of "covariant" term. If 3D vector is not covariant then its components depend on sensor position only. Covariant vector components depend on both orientation and position. Values of components are projections of geometric vector to sensor's axes of reference. Figure 12 presents two orientations of sensor: blue and green. Projections of field vector **A** are different for these different orientations. So we have 3D covariant physical field. The **Sensor** object is a sensor of the field. The **Scalars** object provides components of field vector obtained by the **Sensor** object. These components are used in following motion equations of **Ball 2**:

$$\dot{x} = V_x;$$

$$\dot{y} = V_y;$$

$$\dot{z} = V_z;$$

$$\dot{V}_x = aE_x;$$

$$\dot{V}_y = aE_y;$$

$$\dot{V}_z = aE_z.$$

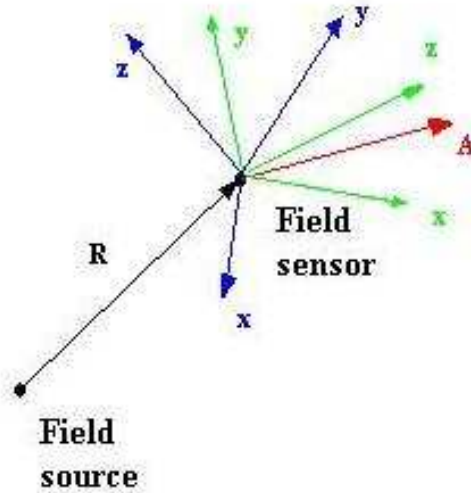


Figure 12: Covariant field

where x, y, z are coordinates of **Ball 2** and V_x, V_y, V_z are components of its velocity. Coefficient a depends of charge of **Ball 2** and system of units. These motion equations are contained in **Motion** object. Then coordinates x, y, z are used by moved reference frame **Sensor's frame**. The **Sensor** object is installed on **Sensor's frame**. So we have backward dependence. Motion and therefore current position of **Sensor's frame** and **Sensor** depends on current field value. Otherwise current field value depends on current position. Let us make situation more complicated. New situation is presented on figure 13 Here we have installed **Field** on **Field's frame** and added new object **Relative**. The **Relative** defines relative motion parameters of **Sensor's frame** with respect to **Field's frame**. Relative distance is indicated by **Chart** object. Physical picture remains the same. We have added indication only. Now let us install **Ball 1** on moved platform as it is presented in figure 14 Simulation of this situation is presented on figure 15.

Now **Field's frame** is moved and **Motion of ball 1** object defines motion low. The time dependence of distance between **Ball 1** and **Ball 2** has been changed by following factors. First of all now **Ball 1** which is source of **Field** is moved. Its own motion influence on relative distance. Secondly motion of field source influence on field intensity near **Ball 2**. So absolute motion of **Ball 2** will be changed. This factor also influence on the distance. These facts are physically evident. I would like to exhibit facilities of such approach from point of view of software user. The user simply has added motion to **Ball 1** and all other dependent factors had been automatically taken into account. This facility is

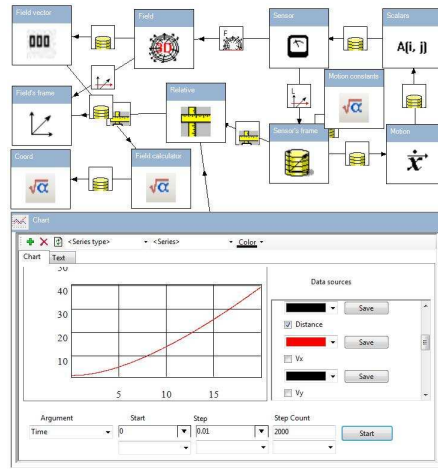


Figure 13: Charged balls motion simulation with relative measurements

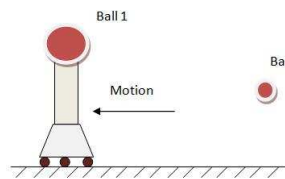


Figure 14: Charged balls with moved ball 1

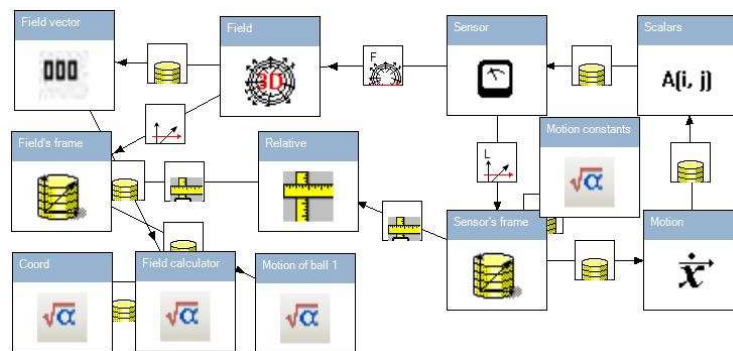


Figure 15: Charged balls motion simulation with moved ball 1

impossible without integration.

4.2 Space aerodynamics and digital image processing

The key feature of space aerodynamics is that spacecraft interacts with molecules which do not collide with each other. Therefore aerodynamic force depends on visible area and does not depend on other parameters of spacecraft shape. We can use digital image processing for space aerodynamic calculation. Let us consider it for space aerodynamics. The image of the Mir orbital station is presented on figure 16. This photo should be filtered for space aerodynamics usage. This

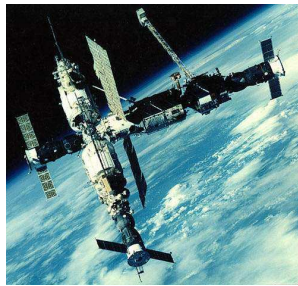


Figure 16: The Mir orbital station

filtering (digital image processing) is presented on figure 17. The **Prototype**

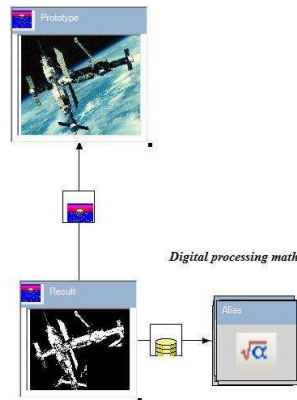


Figure 17: The Mir orbital station digital image processing

object contains source image and **Result** contains filtering result. Other objects contain necessary math. This sample presents main advantage of integration. Software devoted to space technology only is not effective without digital image

processing. Otherwise digital image processing is not effective without advanced math.

4.3 Algebraic topology

Algebraic topology is not yet everyday tool of engineer or scientist. Now this branch of science looks rather as exotic. However exotic tasks are good tests for integrability and universality. The "Modern math" has been developed as test of prospects of Category Theory approach. In this chapter calculation of different topological invariants [2] is considered. Let us consider Klein bottle K and real projective space $\mathbb{R}P^3$ (figure 18 Homology groups of these spaces can

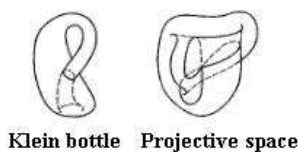


Figure 18: Klein bottle and projective space

be calculated as homology of following chain complexes [2]:

$$\dots \xrightarrow{\partial_3} C_2 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0;$$

Chain complex of Klein bottle K and real projective space is presented below:

$$\mathbb{Z} \xrightarrow{0} \mathbb{Z} \xrightarrow{2} \mathbb{Z} \xrightarrow{0} \mathbb{Z}$$

The application representation of it is shown in figure 19

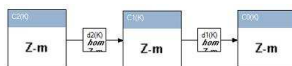


Figure 19: Klein bottle chain complex representation

Chain complex of real projective space $\mathbb{R}P^3$ is presented below:

$$\mathbb{Z} \xrightarrow{\begin{pmatrix} 1 \\ 1 \end{pmatrix}} \mathbb{Z} \times \mathbb{Z} \xrightarrow{(1,1)} \mathbb{Z}$$

Chain complexes of both spaces are presented in figure 20.

Now we can calculate homology and cohomology groups, homology with coefficients and other invariants. More details you can find at Category Theory project <http://categorytheory.sourceforge.net/>.

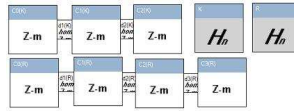


Figure 20: Klein bottle chain complex representation

5 Realistic samples

Here we consider more complicated samples which are related to real engineering problems. This problems include following disciplines:

- advanced mechanics;
- processing of signals;
- statistics;
- system identification;
- control theory;
- celestial navigation;
- astronomy;
- geomagnetism;
- space aerodynamics;
- digital image processing;
- virtual reality. All these samples can be downloaded from <http://www.codeproject.com/KB/architecture/grandiose2.aspx>

5.1 Advanced mechanics

Space technology provides good samples of advanced mechanics. Orbital station (figure 16) is a very complicated mechanical object. It is not rigid body. It has solar cell panels. These panels are elastic. The station is stabilized by gyros. Moreover station configuration is not constant. The "Mechanical aggregate" library had been developed for simulation of similar aggregates. The library contains aggregate designer. Why aggregate designer? Indeed mechanical equations are well known long time ago. But software development for simulation of complicated mechanical objects is not quite easy task. Aggregate designer make this task much easier. Aggregate designer is integrated into framework. This fact enables us provide interpretability of mechanics with physical fields. So it is easy to simulate action of magnetic fields on mechanical objects. I will consider this task below. Now we would like to create mechanical model of spacecraft from models of its modules. Typical spacecraft module is schematically presented in figure 21 This module has own coordinates system $OXYZ$. Also it has places of connections. We can connect other modules to this module. Behavior of module is defined by following kinematic parameters:

- radius vector r ;
- velocity V ;
- orientation quaternion Q ;

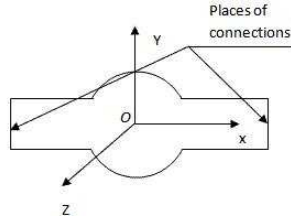


Figure 21: Spacecraft module

- angular velocity ω .

But module is not rigid in general. And these parameters are not parameters of module. These parameters are rather parameters of one point of module. In this article we suppose that these parameters are parameters of origin of $OXYZ$ coordinates system. Since module is not rigid it has additional degrees of freedom. These degrees of freedom can be interpreted as generalized coordinates q_i ($i= 1, \dots, n$). Instant state of module is defined by following parameters: r , Q , q_1, \dots, q_n , V , ω , $\dot{q}_1, \dots, \dot{q}_n$. This parameters will be named state variables. Parameters \dot{V} , $\dot{\omega}$, $\ddot{q}_1, \dots, \ddot{q}_n$ will be called accelerations. Mechanical equations define accelerations by state parameters. Accelerations near connection can be defined by following way:

$$\dot{\omega}_i = \varepsilon_i + P_{\omega_i} \dot{V} + Q_{\omega_i} \dot{\omega} + \sum_k R_{\omega_{ik}} \ddot{q}_k; \quad (1)$$

$$\dot{V}_i = a_i + P_{V_i} \dot{V} + Q_{V_i} \dot{\omega} + \sum_k R_{V_{ik}} \ddot{q}_k. \quad (2)$$

where i is number of connection. Other variables are matrixes which depend on state variables. Connection of two modules is presented on figure 22. Both

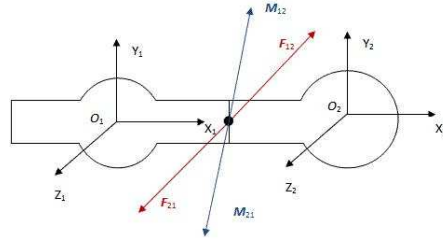


Figure 22: Connection of two modules

modules have equal acceleration near connection. First module acts to second one by force F_{12} and mechanical momentum M_{12} . Similarly second module acts to first one by force F_{21} and mechanical momentum M_{21} . These forces and

momentums satisfy following Newton equations:

$$F_{21} = -F_{12}; \quad (3)$$

$$M_{21} = -M_{12}. \quad (4)$$

Mechanical equations of module can be represented by the following way:

$$\dot{V} = a + \sum_i D_{V_i} F_i + \sum_i E_{V_i} M_i; \quad (5)$$

$$\dot{\omega} = a + \sum_i D_{\omega_i} F_i + \sum_i E_{\omega_i} M_i; \quad (6)$$

$$\ddot{q}_k = a + \sum_i K_{ki} F_i + \sum_i L_{ki} M_i. \quad (7)$$

In these expressions accelerations are independent variables. F_i (M_i) is force (mechanical momentum) of i - h connected module. Other vector and matrix parameters depend on state variables. Suppose that m - h connection place i -h module is connected to n - h connection place j -h module. Then we have following evident relations:

$$\dot{V}_{im} = \dot{V}_{jn}; \quad (8)$$

$$\dot{\omega}_{im} = \dot{\omega}_{jn}; \quad (9)$$

$$F_{im} = -F_{jn}; \quad (10)$$

$$M_{im} = -M_{jn}. \quad (11)$$

Expressions (1) - (11) are linear by accelerations system of equations. This system is fully defined and therefore can be solved. In result we have mechanical equations of whole aggregate. It is worth to note that this system is redundant. If n_1 (n_2) is number of freedom degrees of first (second) module then system has $n_1 + n_2$ degrees of freedom. However aggregate has $n_1 + n_2 - 6$ degrees of freedom. The exist software version that avoid this redundancy. But I will not describe it in this article. Aggregate designer provides equations of full aggregate automatically. Coefficients of (1) - (11) equations are inputs of aggregate designer. In general these coefficients can depend on states of modules. There exist a lot of variants. But we can abstractly define calculation of these coefficients and then provide different implementations. Following types of mechanical modules are implemented:

- absolutely rigid body;
- elastic console;
- flywheel;

Equations of rigid body are well known and are not present them here. In context of aggregate designer rigid body has following properties:

- mass m ;
- momentum of inertia J ;
- number of connections;

- positions and orientations of connections.

These properties can be edited. Elastic console body is a mechanical system of infinite degrees of freedom. Usually math model of this object contains finite degrees of freedom with finite set of valuable harmonic oscillations ???. Every harmonic oscillation can be described by following second order system of ordinary differential equation:

$$A\ddot{q} + \varepsilon\dot{q} + cq = Q.$$

Where Q is generalized force, A , ε and c are coefficients. Number of harmonics and their properties can be edited.

Flywheels are used in spin stabilization systems of spacecrafts. Following documents contain information devoted to spin stabilization systems:

- <http://www.freepatentsonline.com/3767139.html>
- <http://www.aiaa.org/content.cfm?pageid=406&gTable=mtgpaper&gID=58741>
- <http://adsabs.harvard.edu/abs/1966CosRe...4..173A>

Flywheel is forced by reversible engine (figure 23).

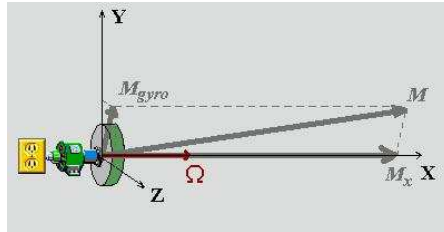


Figure 23: Flywheel

Otherwise flywheel acts to engine by momentum M_x . Since engine is attached to spacecraft this momentum is transferred to spacecraft. This momentum is used for spacecraft stabilization. Since flywheel is rotated we have additional gyro momentum M_{gyro} . Gyro momentum can be calculated by following expression:

$$M_{gyro} = J_F \Omega \times \omega.$$

where J_F is inertial momentum of flywheel, Ω is angular velocity of flywheel and ω is angular velocity of engine (and also spacecraft). Total momentum M is equal to geometric sum $M = M_x + M_{gyro}$; Gyro momentum is undesirable factor. Stabilization system should require following condition $|M_{gyro}| \ll |M_x|$. However since engine acts to flywheel value of is being increased by the time. Increasing of M_{gyro} compensated by other devices which acts to spacecraft. In this article electromagnetic devices will be considered.

Now we can assembly simulation model of spacecraft presented on figure 24. Numbers 1 - 5 are numbers of connections places of spacecraft. Flywheels at-

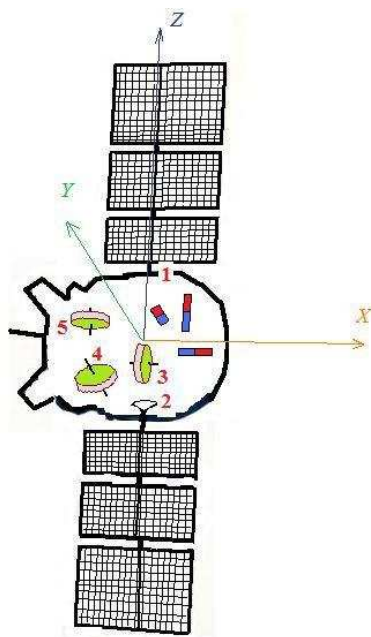


Figure 24: Spacecraft with two elastic consoles and three flywheels

tached to 3, 4, 5 connection places realize angular stabilization of spacecraft with respect to axes X , Y , Z . Besides flywheels spacecraft contains three electromagnets for stabilization. Mechanical model of spacecraft is presented on figure 25. Numbers 1-5 of links are numbers of spacecraft connections.

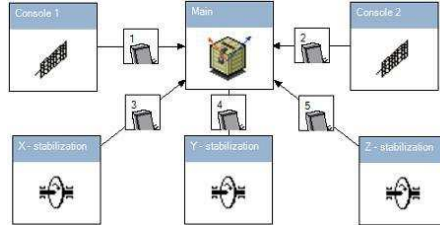


Figure 25: Simulation model of spacecraft with two elastic consoles and three flywheels

5.2 Virtual vibration test

Now we have got simulation model of spacecraft. However this model is not facile for development of stabilization system. Control systems theory usually uses linearized models. Such model can be obtained by virtual vibration test. Note that presented on figure 25 is not facile. It contains a lot of big squares. Presented software supports compact representation of models. So we can replace model (figure 25) by its compact representation (figure 26) During virtual

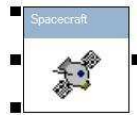


Figure 26: Compact representation of spacecraft simulation model

vibration test or virtual spacecraft is being forced by momentum which satisfies PID control law [4] of momentum has been used:

$$M_x = M_0(t) + K_1\omega + K_2\varphi + K_3 \int \varphi d\varphi.$$

Where M_x is mechanical momentum, is X - projection of spacecraft angular velocity, of spacecraft, is rotation angle of spacecraft with respect to X - axis. Parameters K_1 , K_2 , K_3 are constants. The test purpose is definition of spacecraft transformation function. Its definition can be obtained by response on harmonic input. The chirp input signal has been used:

$$M_0(t) = C \sin(at + bt^2).$$

Virtual vibration test scheme is presented on figure 27 The scheme contains

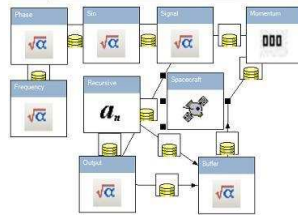


Figure 27: Virtual vibration test scheme

spacecraft's mechanical model and additional math which is necessary for virtual vibration test. In result of vibration test we have obtained response that is presented on figure 28.

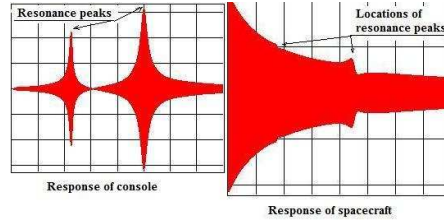


Figure 28: Virtual vibration test scheme

5.3 Nonparametric identification

Nonparametric identification methods include definition of gain response and frequency response. Universal software can easy resolve these tasks. Full explanation reader can find article devoted processing of signals. Results of digital processing is presented in figure 29.

5.4 Parametric identification

Parametric identification methods include definition of transfer functions. Control systems specialists use logarithmic scale for frequency response. This function provides clear picture of control object. So we transform functions of previous sections to logarithmic scale. Logarithmic transformations of gain response a frequency response (figure 29) are presented on figure 30. The Y - axis of frequency response is also logarithmic. Control system specialist can define that such charts correspond to following transfer function:

$$W(s) = \frac{k T_1 s^2 + T_2 s + 1}{s T_3 s^2 + T_4 s + 1} \frac{T_5 s^2 + T_6 s + 1}{T_7 s^2 + T_8 s + 1}$$

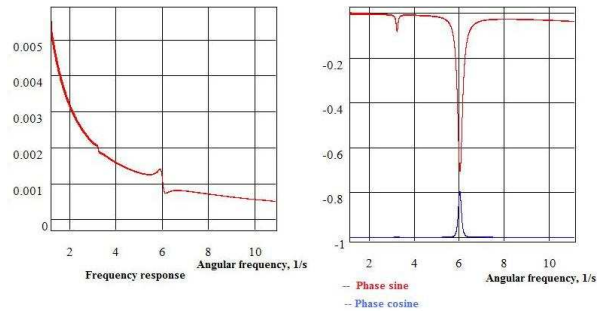


Figure 29: Result of digital processing of virtual vibration test signals

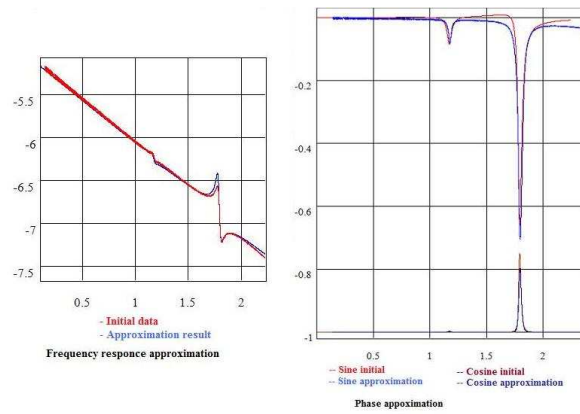


Figure 30: Approximation results

But parameters k, T_1, \dots, T_8 are unknown. These parameters can be defined by nonlinear regression. Regression algorithm is presented in figure 31. Charts in

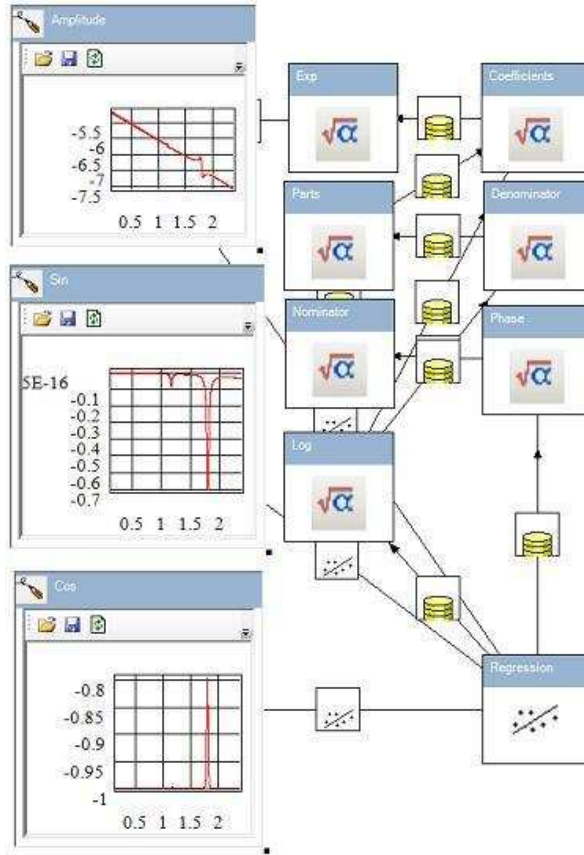


Figure 31: Regression algorithm of parametric identification

the left part of figure 31 represents approximated functions (Frequency response, sine and cosine of phase). Other squares contain necessary math. Approximation result is presented in figure 30.

5.5 Celestial navigation

Any control system requires sensor. We use celestial navigation sensor for spacecraft control. The sensor enables to define orientation of spacecrafts. There are a lot of types of celestial navigation sensors. Here one of possible schemes is provided. Suppose that we have equipment that provides celestial images and star catalogues. Comparison of image and catalogue enable us to define orien-

tation of equipment. So we can define orientation of spacecraft. Algorithm of this sensor is presented in figure 32. First of all let us consider image process-

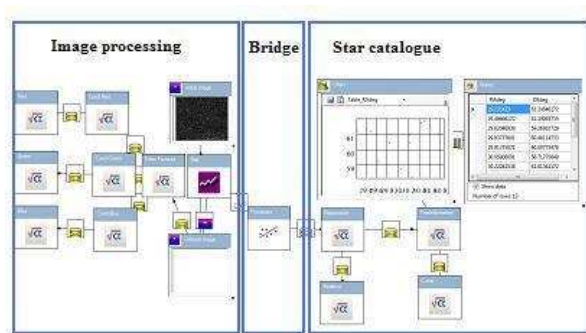


Figure 32: Celestial navigation algorithm

ing. Any celestial image contains interfering information. We need filtration for its exclusion. Nonlocal digital image processing is being used for this purpose. This scheme (figure 32) contains **Initial image** (Source image) obtained by equipment. Little squares in figure 32 provides necessary math. It result we have **Filtered image** (Filtration result). Both images are presented in figure 33. Figure 34 explains image filtration algorithm. If we have 9 closed white pix-

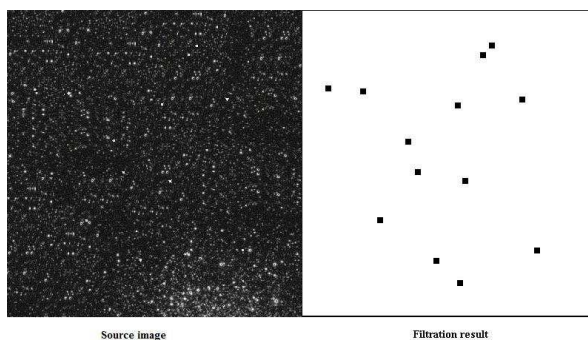


Figure 33: Image filtration in celestial navigation algorithm

els then we replace it by one black pixel. Every other pixels are white. Result of filtration enables us to obtain X and Y coordinates of black pixels. Then this numbers will be compared with star catalogue. The **Stat** component extracts this numbers from **Filtered image**. Star catalogue is stored in database. Necessary information can be extracted by SQL query. Query statement is presented below:

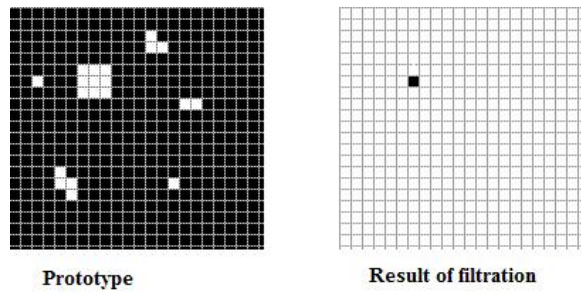


Figure 34: Image filtration in celestial navigation algorithm

```
SELECT RAdeg, DEdeg FROM hip_main WHERE RAdeg > @RAMIN AND RAdeg < @RAMAX AND
@DEMIN AND DEdeg < @DEMAX AND BTmag > @BTMIN ORDER BY RAdeg
```

This statement has following meaning. First of all we consider limited area of sky. Declination and right ascension belong to small intervals. Secondly we consider only such stars which magnitudes exceed defined constant (in this sample the constant is equal to 9). Query result provides following chart presented in figure 35.

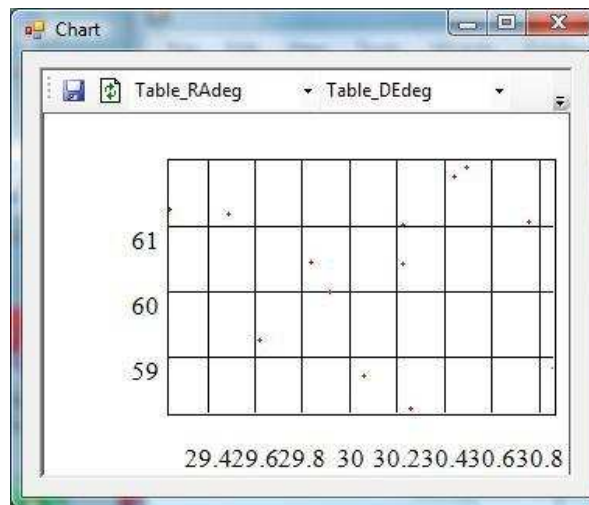


Figure 35: Star catalogue query chart

We would like compare this chart with filtered image. This operation requires a set of math transformations. Essential feature of these transformations is

Euclidean transformation:

$$x' = (x + a) \cos \varphi + (y + b) \sin \varphi;$$

$$y' = -(x + a) \sin \varphi + (y + b) \cos \varphi;$$

Parameters a , b , and φ are unknown. Comparison of star catalogue and filtered image enable us to define these parameters. Using these parameters we can define orientation of spacecraft.

5.6 Motion of mass center

Since we consider magnetic method gyro momentum of compensation we should have model of Earth's magnetic field. But Earth's magnetic field induction depends near spacecraft depends on spacecraft position. So we need consider motion of spacecraft as a point. Here I consider usage of inertial reference frame. It is more convenient for some tasks. Relation between these reference frames is shown in figure 36

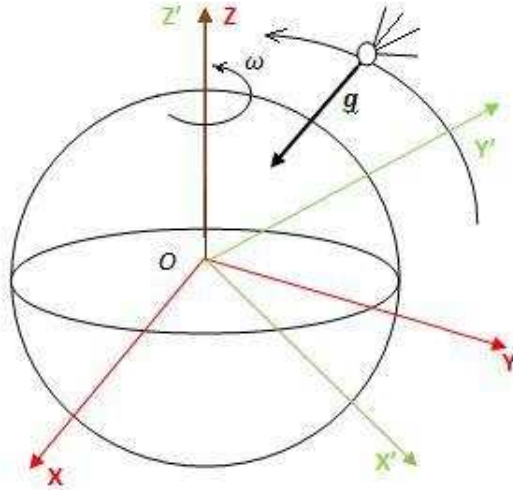


Figure 36: Inertial reference geometry

The $OXYZ$ is inertial frame and $OX'Y'Z'$ is Greenwich one. Greenwich frame is rotated with respect to inertial one. We should transform acceleration vector \mathbf{g} from Greenwich frame to inertial frame. Calculation of \mathbf{g} in inertial reference frame has two features. Satellite coordinates with respect to $OXYZ$ are not equal to coordinates with respect to $OX'Y'Z'$. Moreover projections of \mathbf{g} on $OXYZ$ axes of coordinates are different to projections on $OX'Y'Z'$ axes of coordinates. Declarative approach enables us to resolve both problems at once.

Usage of covariant fields provides solution of both problems Simulation of linear satellite motion is presented in figure 37.

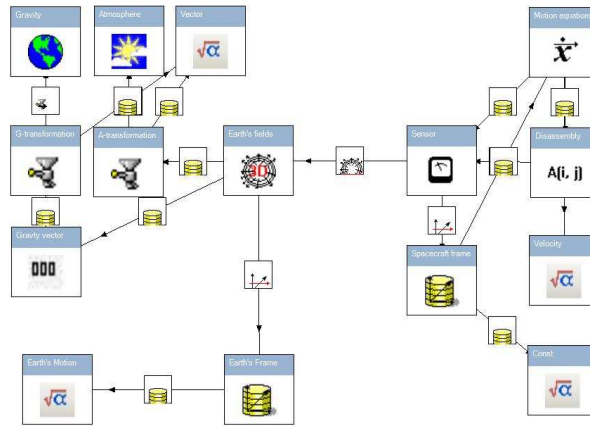


Figure 37: Inertial reference frame geometry

Here **Earth's Frame** component represents Greenwich reference frame. The **Gravity** component is obtained from dynamically linked library. This component calculates Earth's gravity field/ Here this component is used as physical field (in Earth's fields). We can say the same about **Atmosphere** which calculates parameters of dynamical atmosphere model. Physical field is linked to Greenwich reference frame. Gravity field is marked as covariant. These circumstances provide solution of both above problems. The Sensor is linked to Spacecraft frame. Its orientation coincides with orientation of inertial reference frame. Sensor results are used in Motion equations of spacecraft. Otherwise Motion equations results are used by Spacecraft frame. So we have math model of spacecraft linear motion.

The picture in figure 37 is not facile for further development since it contains a lot of squares. Some of squares should be encapsulated. Encapsulation means that we make invisible some elements in figure 37 Now we would like add model of Earth's magnetic field. Result of this operation is presented in figure 38 We

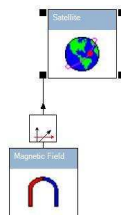


Figure 38: Encapsulated of spacecraft motion model and magnetic field model

have full mechanical model. Next step is development of stabilization system. Development of stabilization system is based on results which was considered in 5.4. Physical principles of control system will be considered in 5.7.

5.7 Final result

Since this article is more illustration of usefulness of advantages of top-down paradigm we leave details of full construction. More details you can find in <http://www.codeproject.com/KB/architecture/grandiose2.aspx>. Figure 39 shows final result. In brief figure 39 contains two main parts. Left part

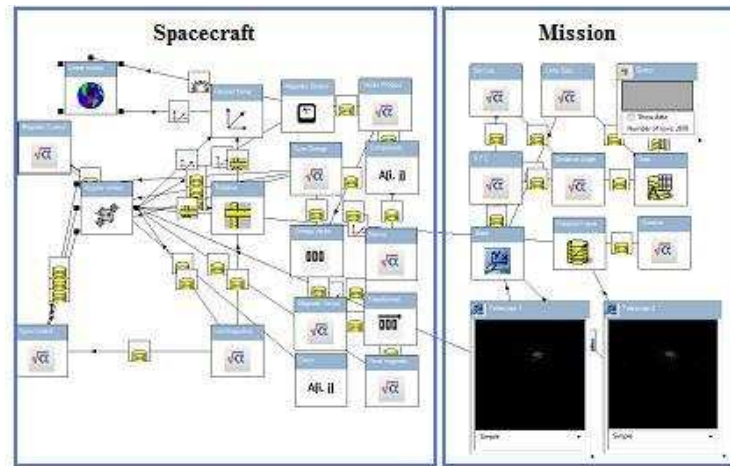


Figure 39: Spacecraft and mission

contains full motion model of spacecraft. Full motion model includes:

- mechanical model of spacecraft;
- model of Earth's gravity field;
- model of Earth's atmosphere;
- model of Earth's magnetic field;
- model of Celestial navigation;
- model of stabilization system of spacecraft.

Stabilization system could not use flywheels only (See 5.1). Otherwise there is an obstacle for construction of stabilization system which uses electromagnets only. Electromagnetic momentum is always perpendicular to magnetic induction B . But stabilization requires all directions of control momentums. So space technology uses double-loop systems which use both flywheels and electromagnets. Let us consider both loops of this system. First loop (I name it high frequency loop) is presented in figure 40 We use celestial navigation for definition of orientation and optical gyroscope for definition of angular velocity. In

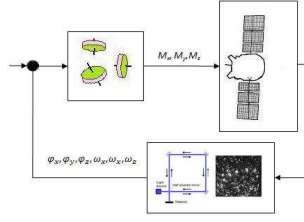


Figure 40: High frequency loop

result we have following parameters $\varphi_x, \varphi_y, \varphi_z, \omega_x, \omega_y, \omega_z$. First three parameters are angle deviations with respect to desired axes X, Y, Z. Following three parameters are angular velocities with respect to same axes. Control momentums are obtained by flywheels. We have identified spacecraft angular motion model in 5.4. In accordance to identified model we develop control law. I drop details here. Here I note that control law of high frequency loop is designed in compliance with control theory. Second loop scheme is presented in figure 41 This loop purpose is limitation of flywheels' angular velocities. Sensor of this

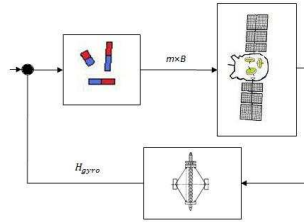


Figure 41: Low frequency loop

loop are tachometers of flywheels. Necessary momentum is provided by electromagnets . It is possible to use different control laws for this loop. But main idea of these laws is single In this article I have used following control law. Let be H_{gyro} is total angular momentum. Then momentum of electromagnets is opposite to H_{gyro} . But electromagnets are not always switched on. If $|H_{gyro}|$ is too small then electromagnets are switched off. Otherwise if angle between Earth's magnetic induction H_{gyro} and H_{gyro} is too small then electromagnets cannot provide substantially large momentum that is opposite H_{gyro} . Therefore if angle between H_{gyro} and B is too small then electromagnets are also switched off. Since action of electromagnets is not continuous I name this loop low frequency loop. Here I explain how magnetic momentum reduces angular velocities of flywheels. Magnetic momentum causes deviation of spacecraft orientation. High frequency loop tries to eliminate this deviation by changing of angular velocity of flywheels. High frequency loop tries to eliminate this deviation by changing of angular velocity of flywheels. Since electromagnetic momentum is

opposite to H_{gyro} changing of angular velocities is reducing of their values. Right part in figure 39 presents spacecraft mission. The mission is astronomical observations. Pure mission is shown in figure 42 Spacecraft contains two tele-

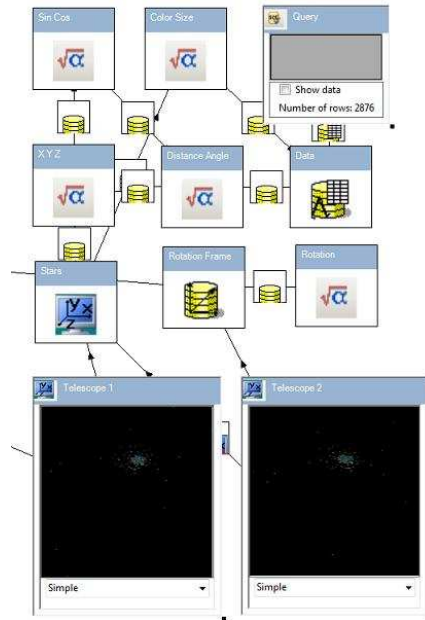


Figure 42: Spacecraft mission

scopes. **Telescope 1** is rigidly attached to spacecraft and do not use additional frame. **Telescope 2** is rotated relatively spacecraft. So **Telescope 2** is installed on **Rotation frame**. Otherwise **Rotation frame** is installed on spacecraft body. Special component has been developed for indication of stars. Properties of the component are presented in figure 43. These properties have following meaning. Star coordinates X, Y, Z are equal to $Formula_1, Formula_2, Formula_3$ (of $X Y Z$ component) respectively. Similarly color and size of star indication are defined. This sample requires Astronomy Express project and star catalogue (Hipparcos and Tycho).

Both telescopes observes single collection of stars (**Stars** component). This collection is generated by following way. First of all **Query** component performs SQL query of star catalogue. Then other components perform necessary math transformations. Result of these transformations is used by **Stars** component.

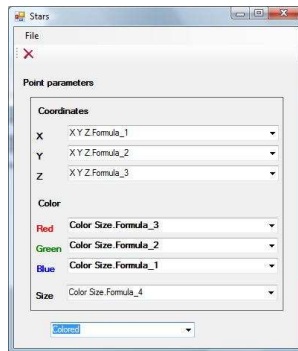


Figure 43: Visualization component properties

6 Conclusion

Typical way of field of activity contains two stages. During first stage there is information acquisition and experience without understanding of fundamental principles. Many almost equivalent results are obtained independently by different researchers and/or developers. Then fundamental principles become well known and chaotic evolution changes to planned development. For example design patterns of GoF [5] had being developed independently. After GoF these fundamental principles of software development became well known. I think that science and engineering software do not reach second stage yet in general. There are good fundamental principles of CAD, CAE etc. But they are not principles of all science and engineering software. I would like to show usefulness of such universal principles.

References

- [1] Robert Goldblatt. Topoi. The categorical analysis of logic. NORTH-HOLLAND 1984.
- [2] Edwin H Spanier. Algebraic topology. McGraw-hill series in higher mathematics 1967.
- [3] Harmonic oscillator. Wikipedia. http://en.wikipedia.org/wiki/Harmonic_oscillation.
- [4] PID controller. Wikipedia. http://en.wikipedia.org/wiki/PID_controller.
- [5] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software (ISBN 0-201-63361-2)