

АВТОМАТИЗАЦИЯ УПРАВЛЕНИЯ ЖИЗНЕННЫМ ЦИКЛОМ ИЗДЕЛИЯ

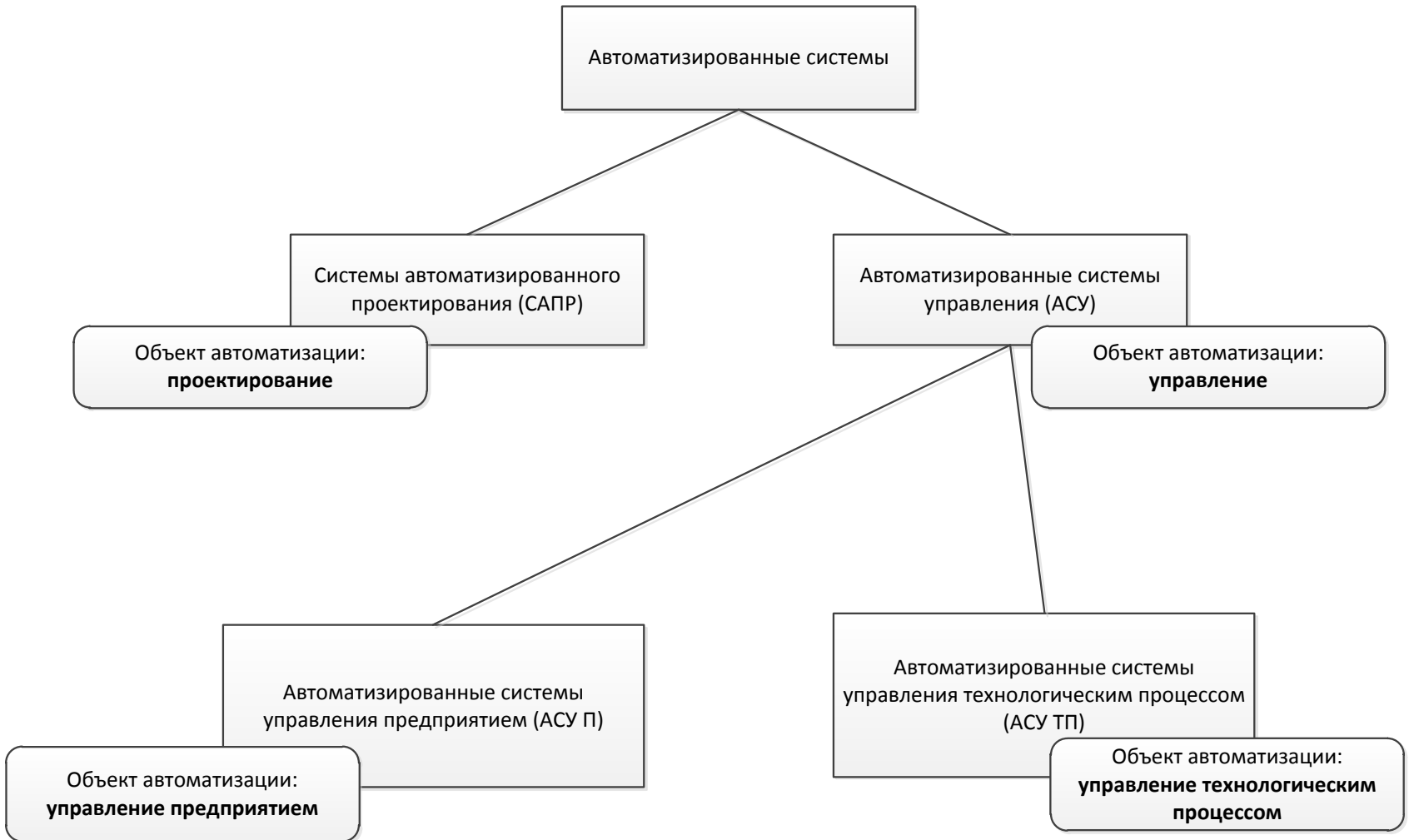
1. Этапы и модели жизненного цикла изделия

- CALS - Continuous Acquisition and Lifecycle Support
- CALS - Computer Aided Acquisition and Logistic Support
- CALS - Computer-Aided Lifecycle Support
- ИПИ - Информационная поддержка процессов жизненного цикла изделий

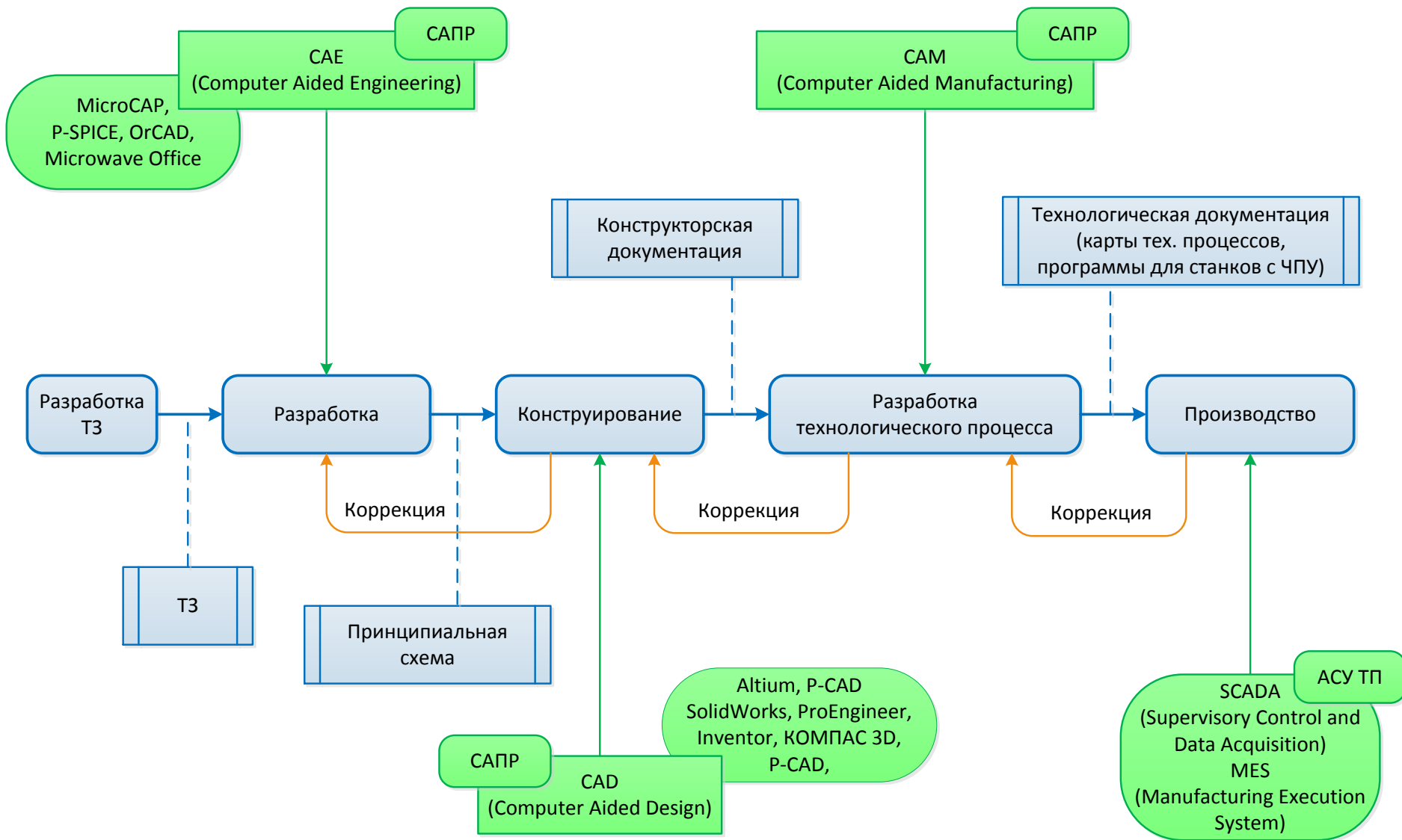
Основные стандарты

- **Р 50-605-80-93** Рекомендации. Система разработки и постановки продукции на производство. Термины и определения
- **ISO 10303 (STEP)** *Standard for Exchange of Product model data*
- **ISO 13584 PLIB** *Industrial Automation - Parts Library*

Типы автоматизированных систем







PCB

3D Models

Apply Clear Zoom Level...

Mask Select Zoom Clear Existing

24 Component Classes (1 Highlighted)

- <All Components>
- <Bottom Side Components>
- <Inside Board Components>
- <Outside Board Components>
- <Top Side Components>
- COM_FLASH
- COM_SDRAM
- COM_SDRAM
- U_1WB_DS2406_EEPROM
- U_Bypass_1V2
- U_Bypass_2V5

164 Components (1 Highlighted)

Designator	Comment	Footprint
U3	S29GL256N11F	BGA100P8X8-6
U4	XC4VX35-10F1	XC4VX25-10F
U5	MAX1831EEE	SOP63P602-16
U6	MAX8860EUA1	SOP63P490-8
U7	SDRAM-16Mx1	TSOP80P1176-
U8	SDRAM-16Mx1	TSOP80P1176-
U9	SRAM-256Kx16	TSOP80P1176-
U10	SRAM-256Kx16	TSOP80P1176-
U11	SRAM-256Kx16	TSOP80P1176-
U12	SRAM-256Kx16	TSOP80P1176-

1 Highlighted Models Hide


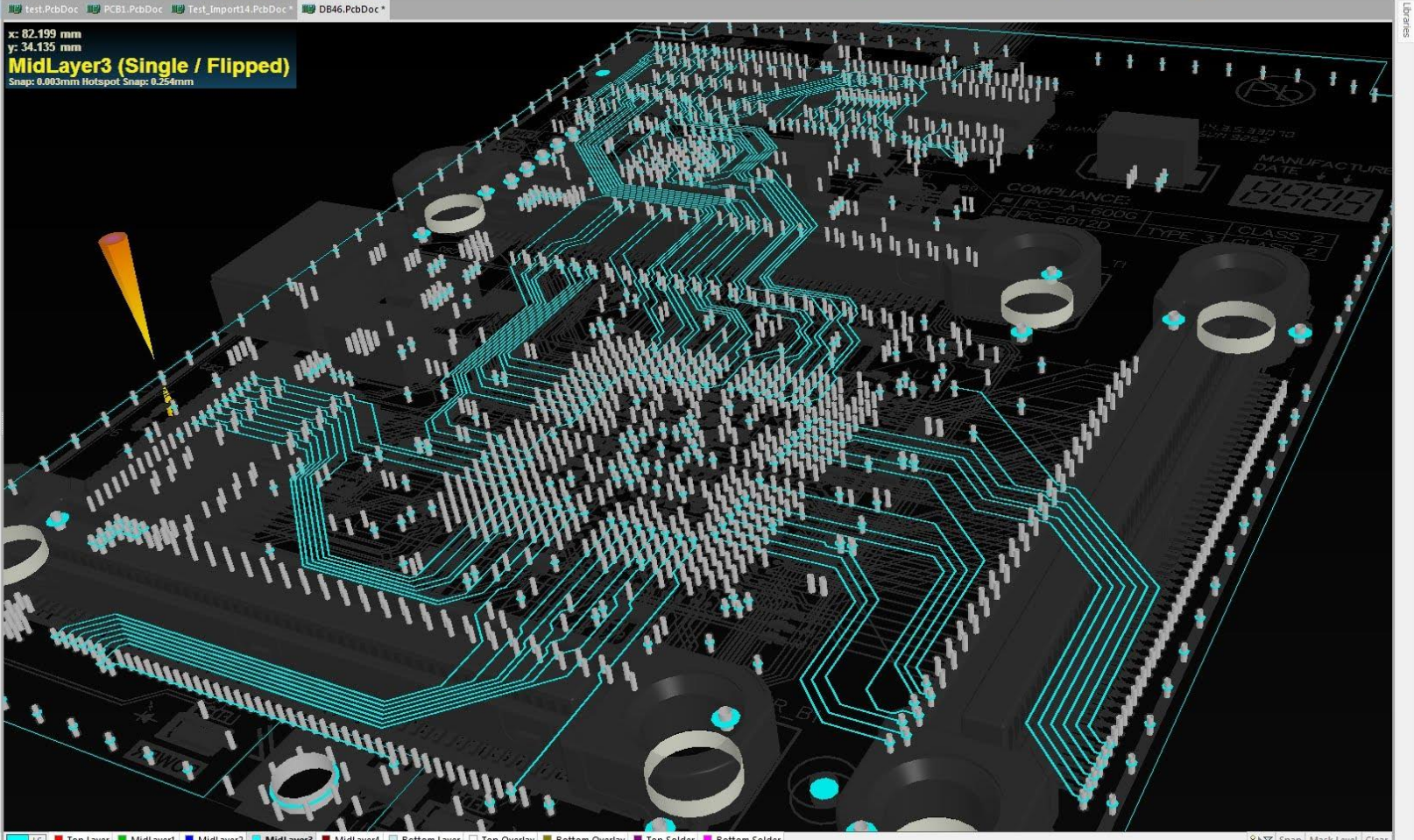
2 Models (2 Highlighted)

Description	Type	DRC
chip only step [U4-3D STEP]		
S135 text [U4-3D STEP] STEP		

2 Highlighted Models Hide

3D Bodies Display Options

- Show Simple 3D Bodies (Using System Setting)
- Show STEP Models (Using System Setting)

Lab1 - Multisim - [Lab1 *]

File Edit View Place MCU Simulate Transfer Tools Reports Options Window Help

--- In Use List ---

Design Toolbox

Lab1
Lab1

V4 15 V
U1 741
C1 47uF
R1 680Ω
R2 680Ω
V5 50mVrms 10000 Hz 0°
R6 5.6kΩ
R5 3.3kΩ
V1 15 V
U2 741
C2 47uF
R3 68kΩ
R4 68kΩ
V2 15 V
V3 15 V

Bode Plotter-XBP1

Mode: Magnitude Phase

Horizontal: Log Lin Vertical: Log Lin

F 20 kHz F 50 dB
I 5 kHz I -0.01 dB

Controls: Reverse Save Set...

15.914 kHz 6.221 dB

Tran: 0.818 s

start
Circuit Design Suite (...)
Lab1 - Multisim - [Lab...]
10:04 PM
Thursday
12/11/2008

Стратегическое управление

ВРМ-системы

Консолидированное управление

Управление финансово-хозяйственной деятельностью ЖКК

ERP-системы

Оперативный мониторинг поставок Энергоресурсов. Контроль качества поставок, формирование энергобалансов. Техническое обслуживание и ремонты. Оперативно-диспетчерская служба

MES-системы

Учет энергопотребления

Контроль аварий

SCADA - системы



Вид

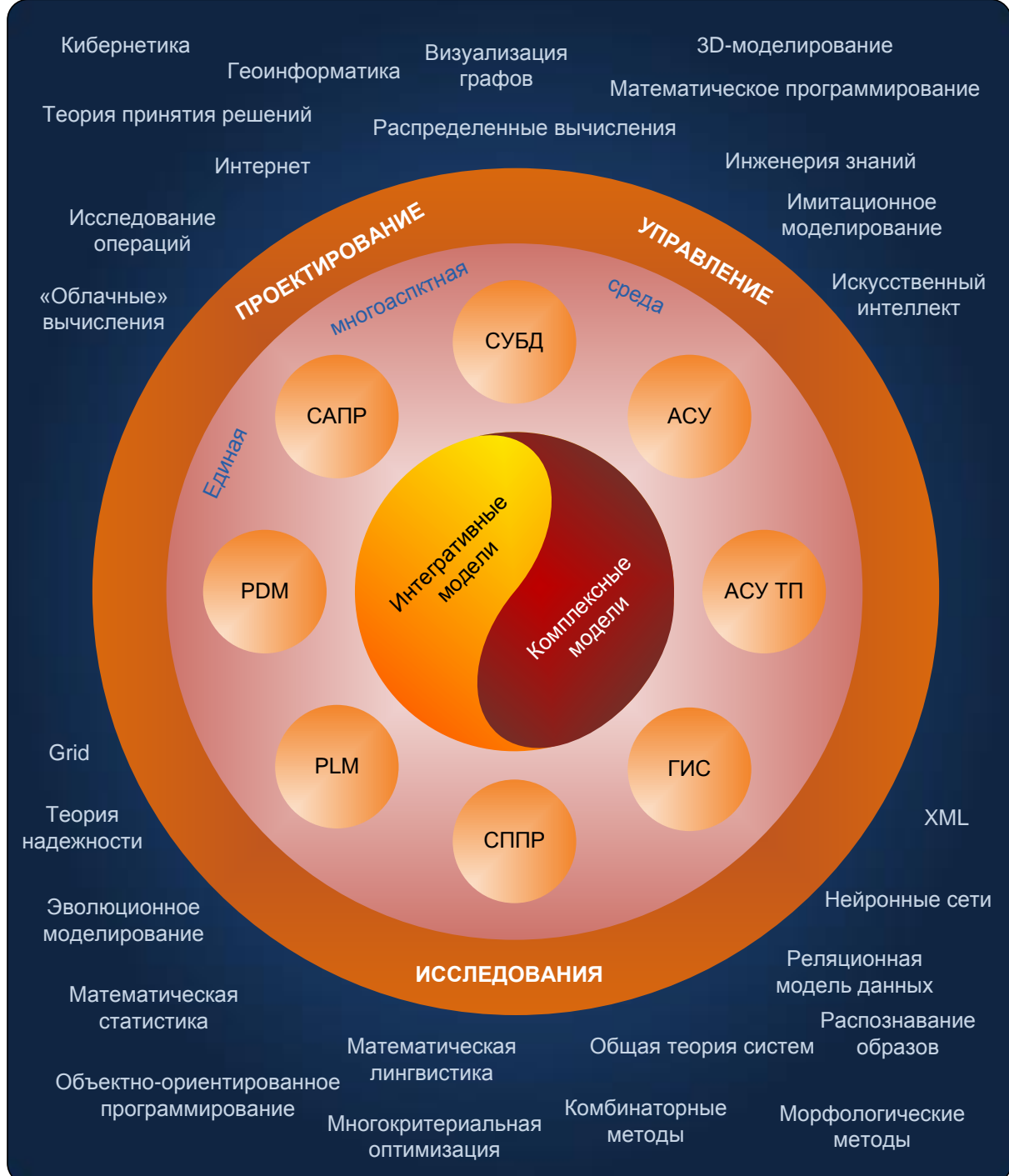
Потребители

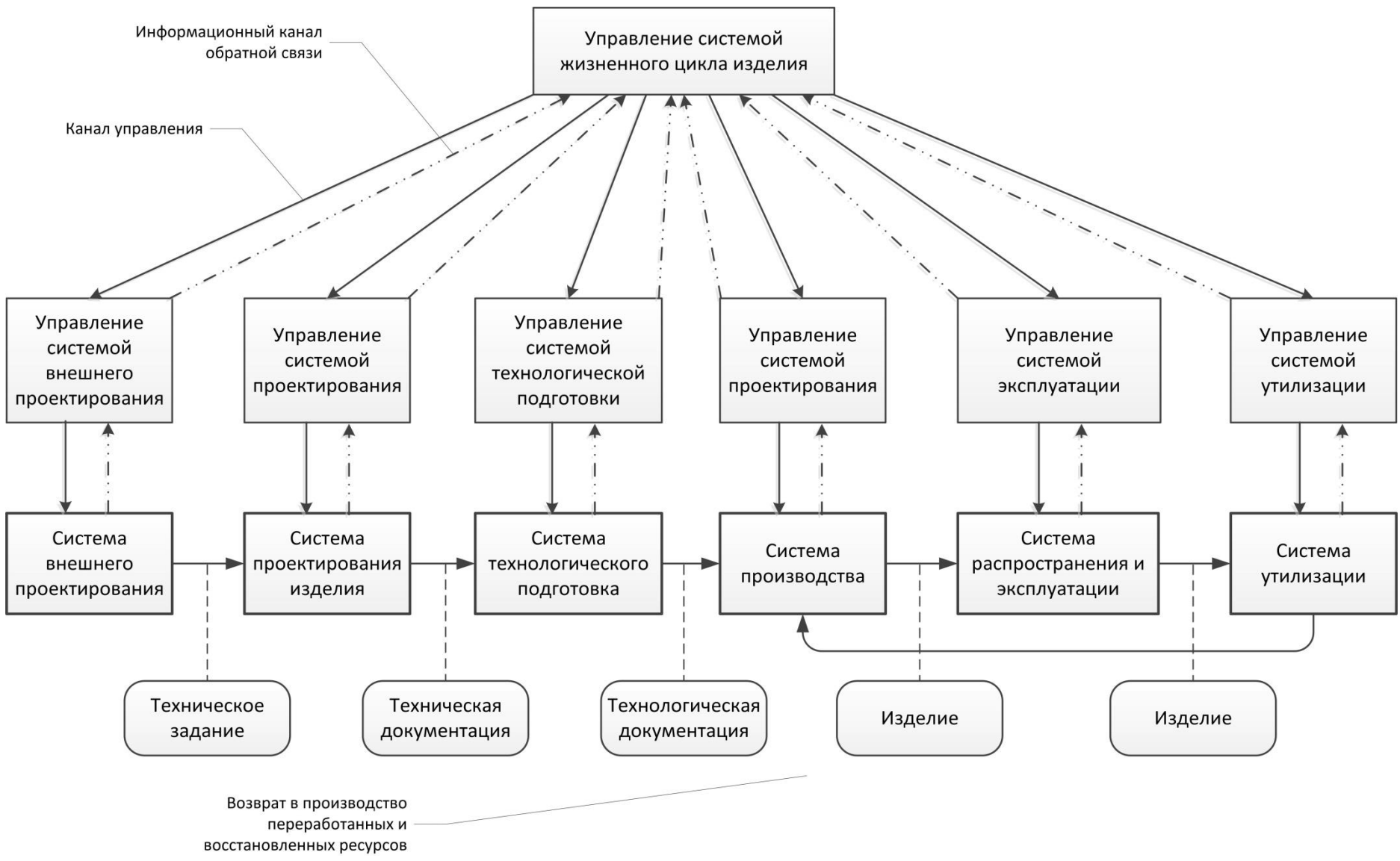
Потребители

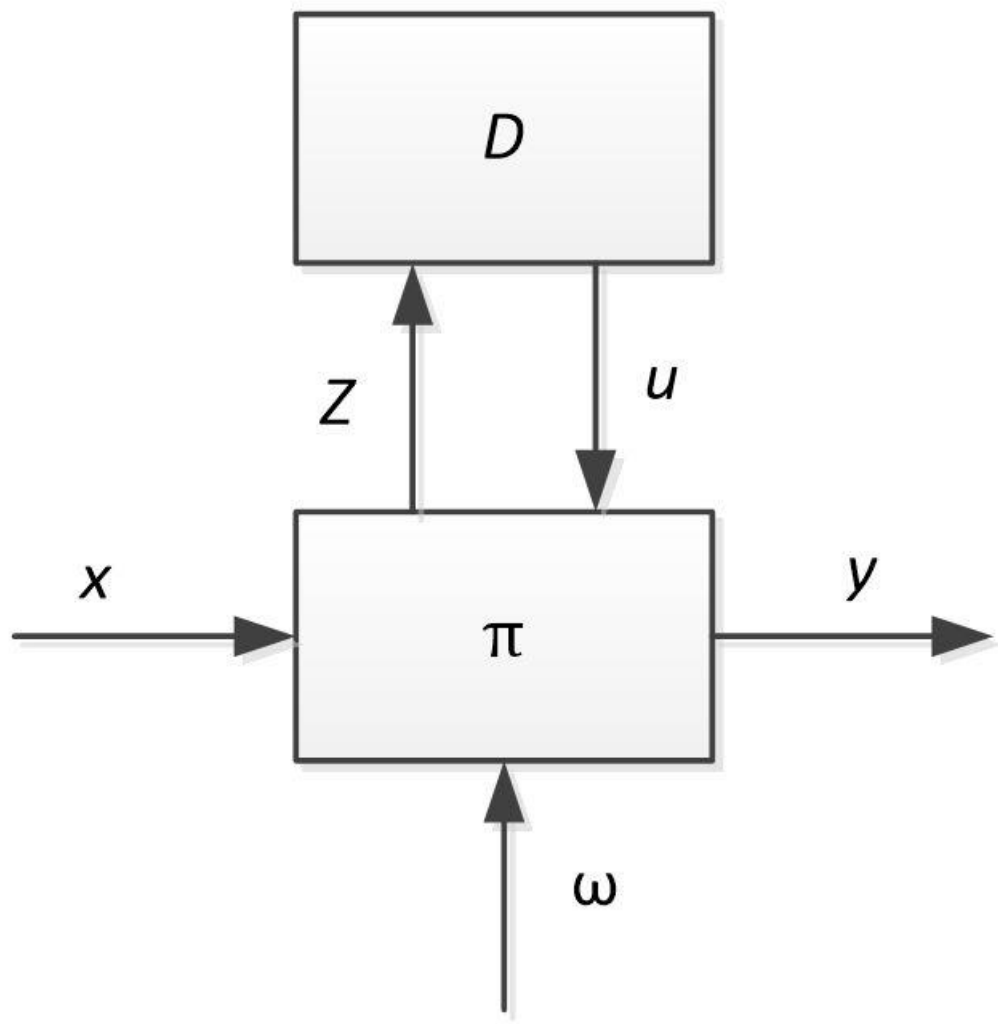
Потребители

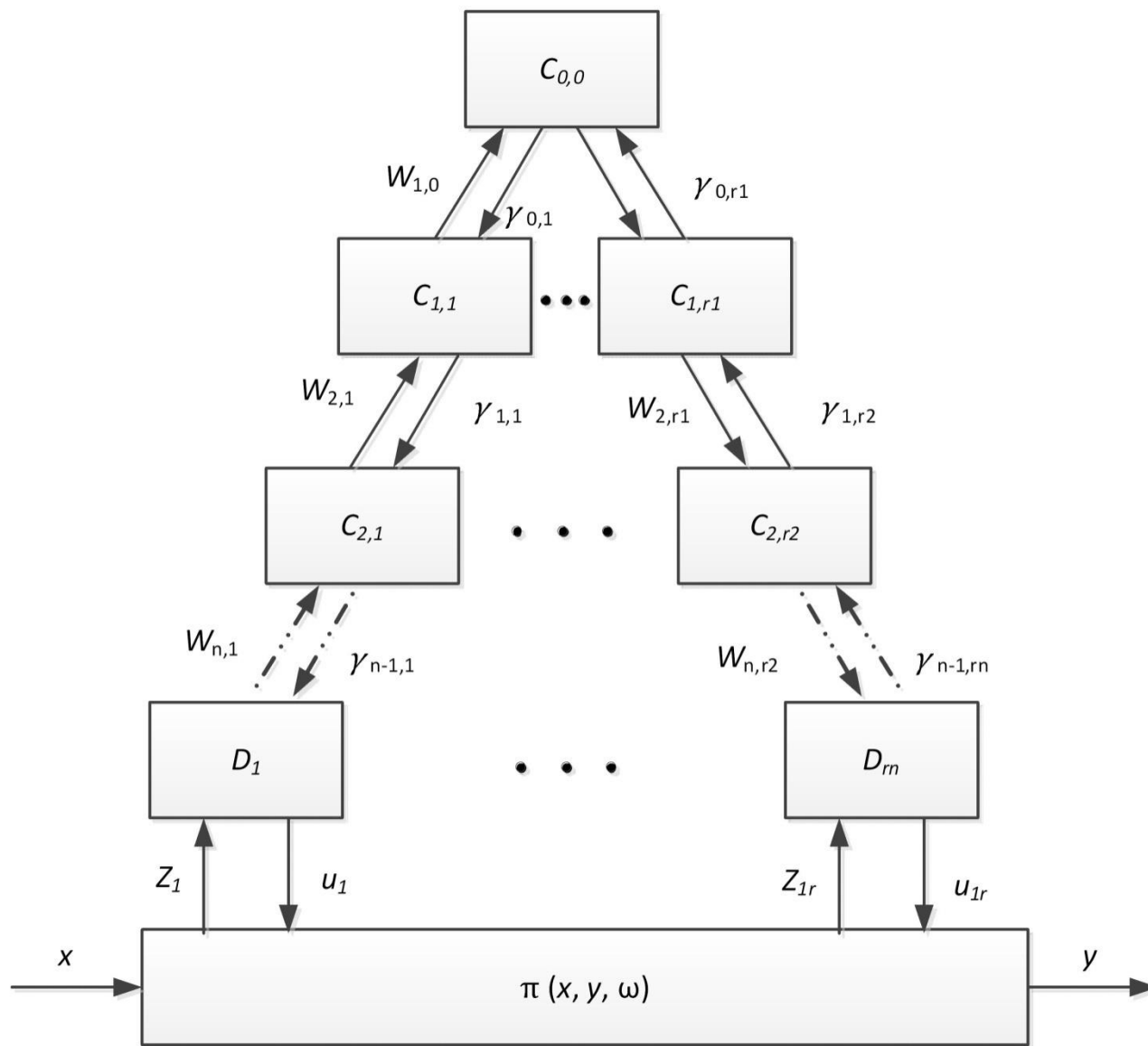
Потребители

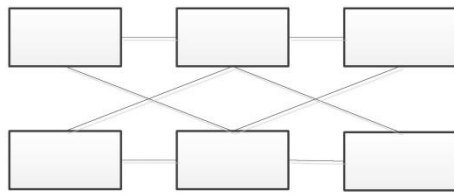
Тип системы	Назначение
CAD	Computer Aided Design. Автоматизация проектирования
CAE	Computer Aided Engineering. Автоматизация инженерных расчетов
CAM	Computer Aided Manufacturing, Автоматизация подготовки технологического процесса производства
PDM	Product Data Management. Автоматизация управления информацией об изделии
PLM	Product Lifecycle Management. Автоматизация управления жизненным циклом изделия.
ERP	Enterprise Resource Planning. Автоматизация управления предприятием
MES	Manufacturing Execution System. Автоматизация управления производственными процессами, включая планирование оптимальной загрузки оборудования
SCADA	Supervisory Control and Data Acquisition. Автоматизация сбора, обработки, отображения и архивирования информации об объекте управления
MRP	Material Requirement Planning. Автоматизация планирования потребности в материалах.
CRM	Customer Relationship Management. Автоматизация взаимодействия с клиентами



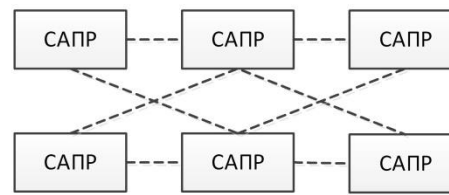




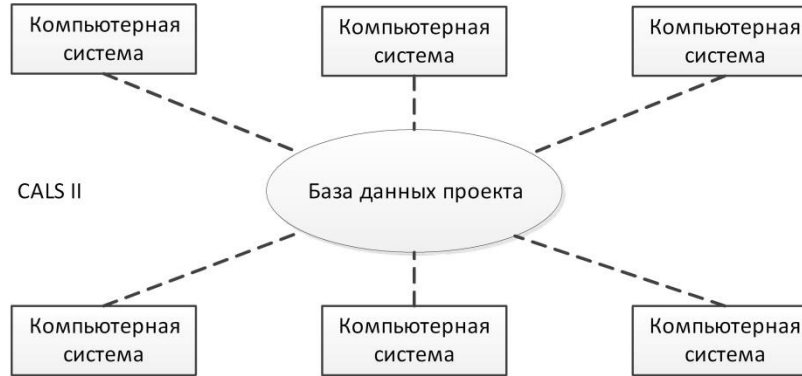




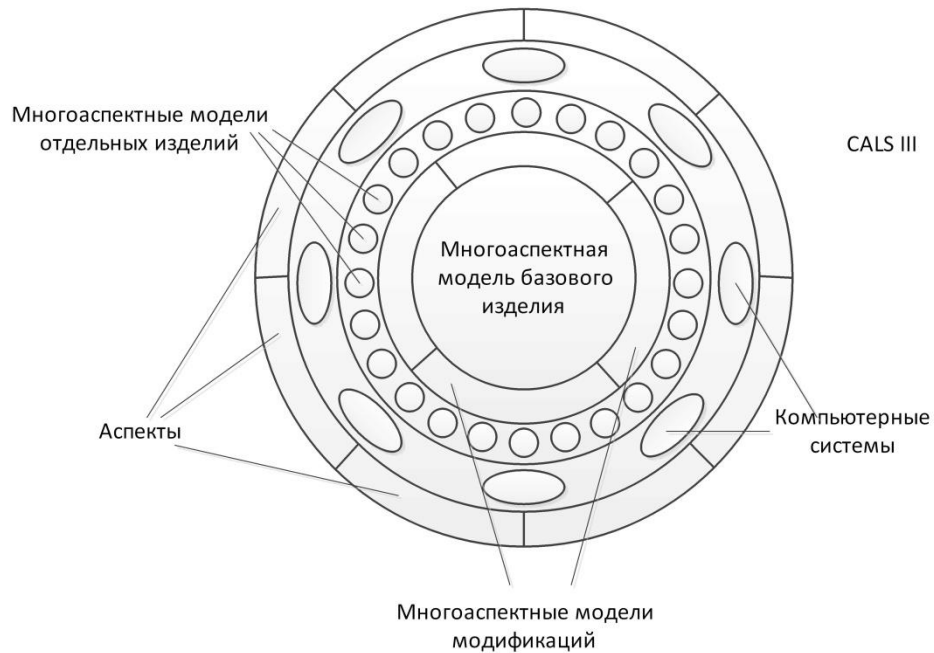
Бумажный документооборот



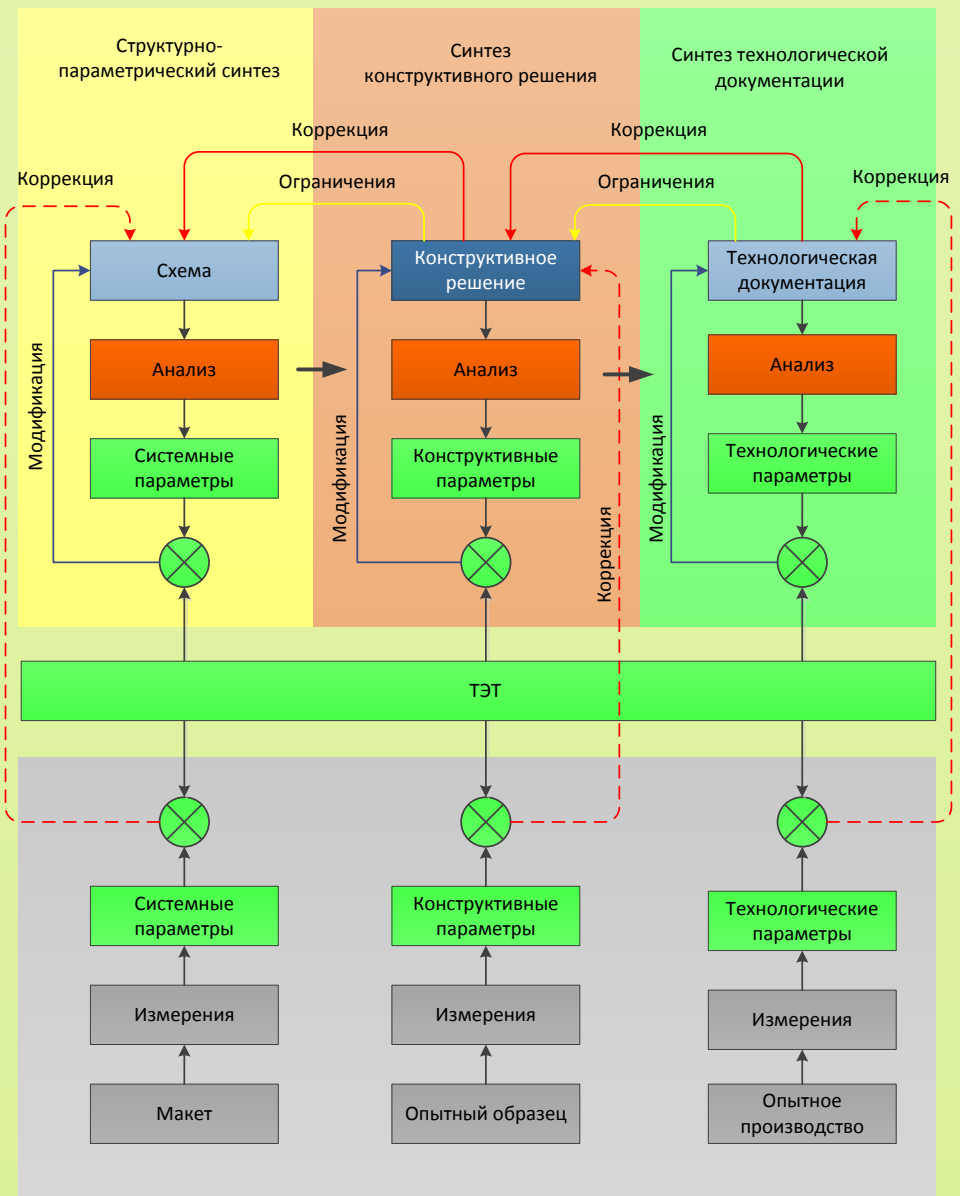
CALS I



CALS II



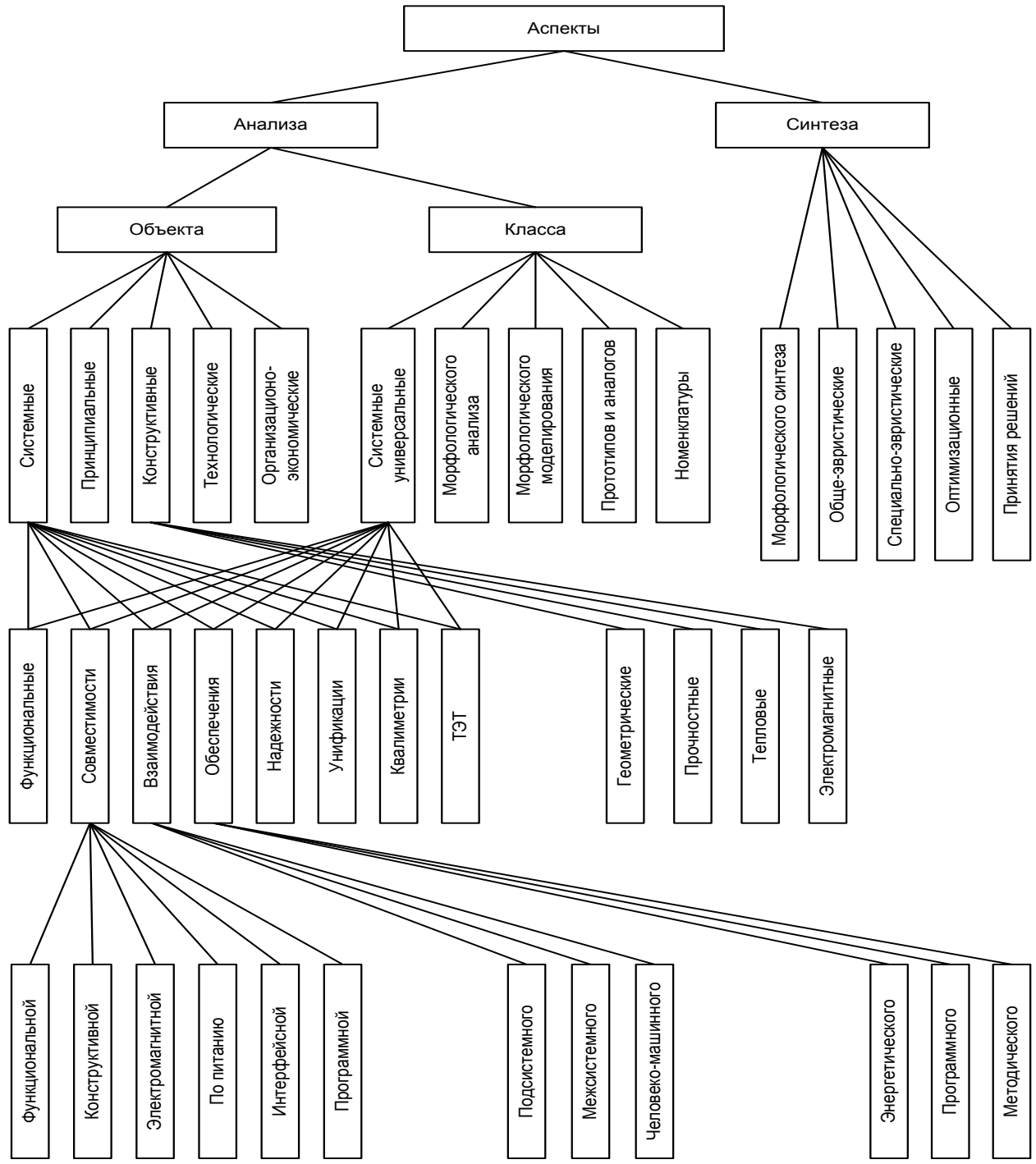
CALS III

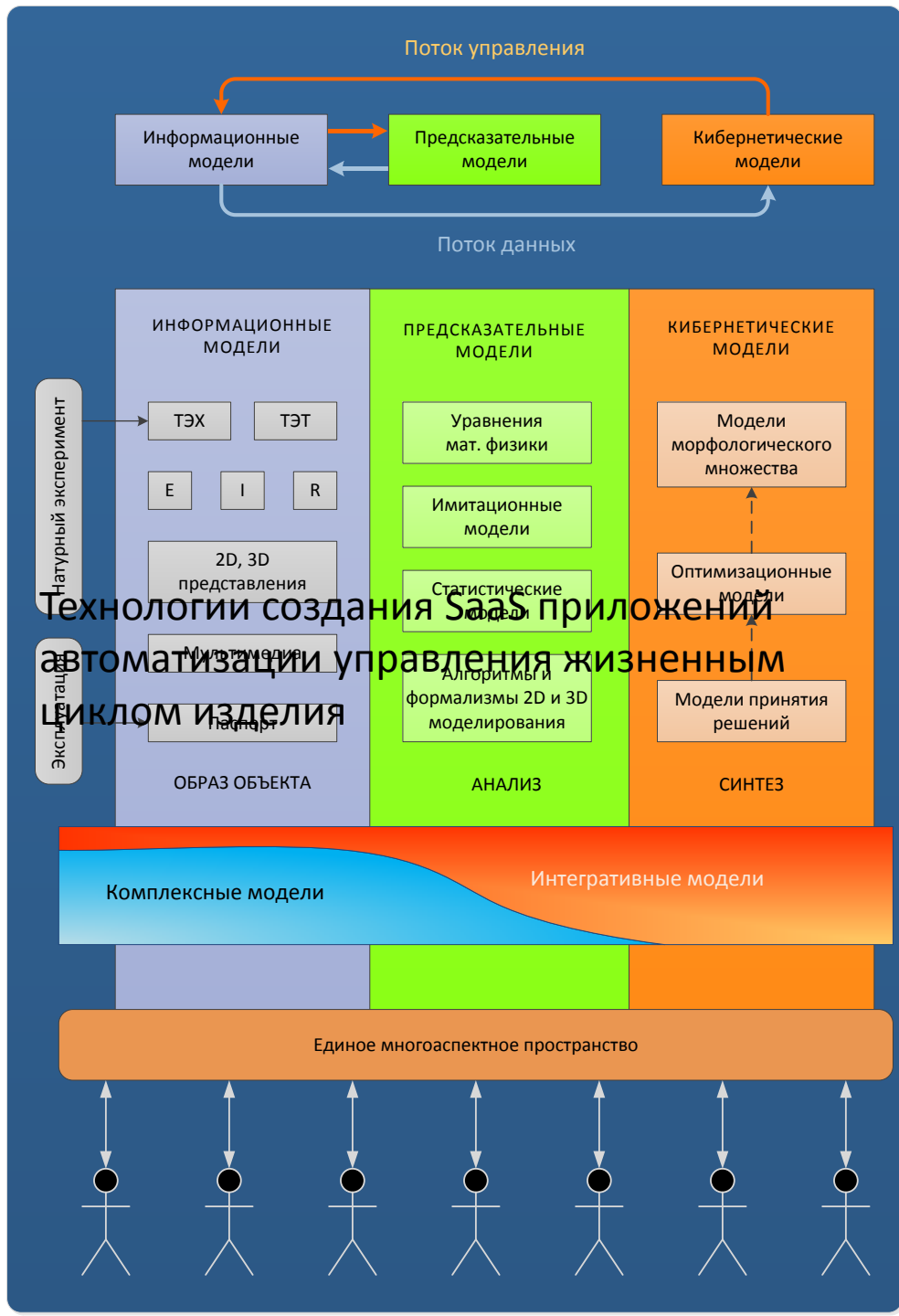


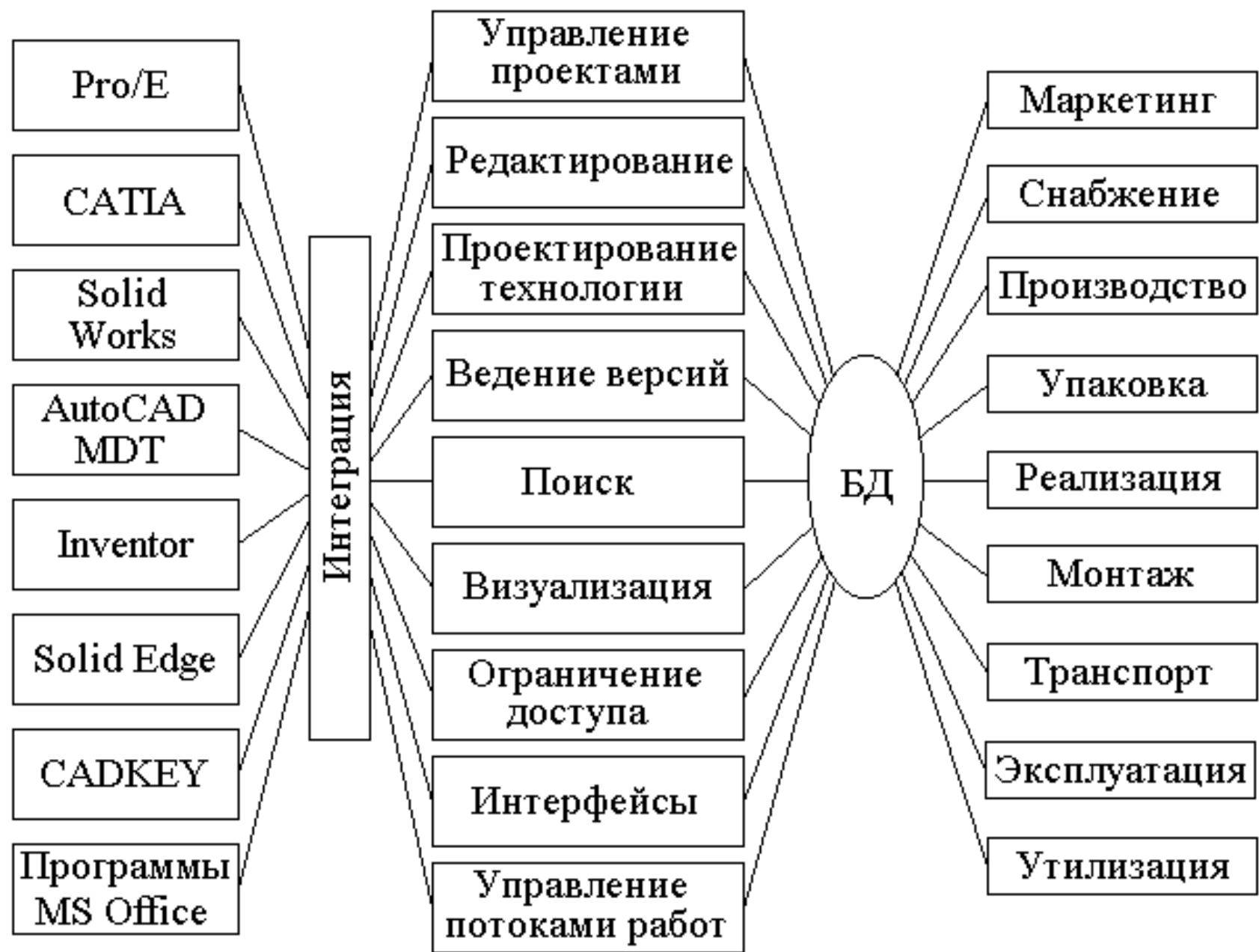
Условные обозначения

Структура $S = \langle E, R \rangle$	Геометрия	Расчеты	Параметры	«Железо»
---	-----------	---------	-----------	----------

2. Информационное обеспечение CALS

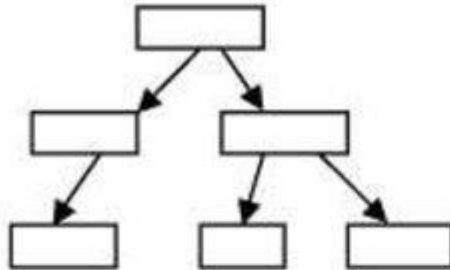






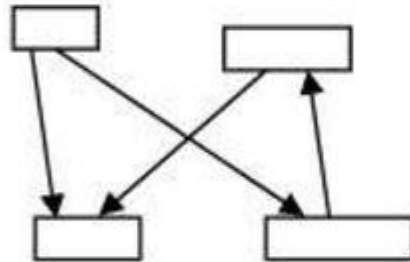
Типы моделей данных

Основные типы моделей данных



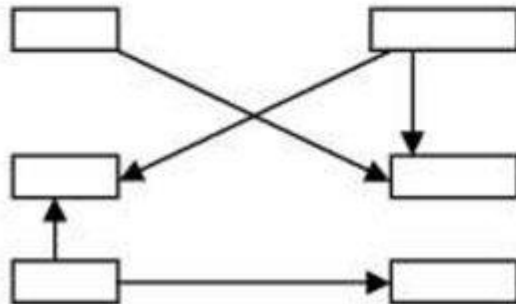
Иерархическая

Взаимосвязи между данными жестко фиксированы.
Изменение связи ведет к реорганизации структуры.
Число связей ограничено.



Сетевая

Характер связей более разнообразен.
Трудно вводить изменения.



Реляционная

Таблицы независимы.
Связи полностью изменчивы.
Простота расширения.

EAV

Entity

Products
Category
Customer
Orders

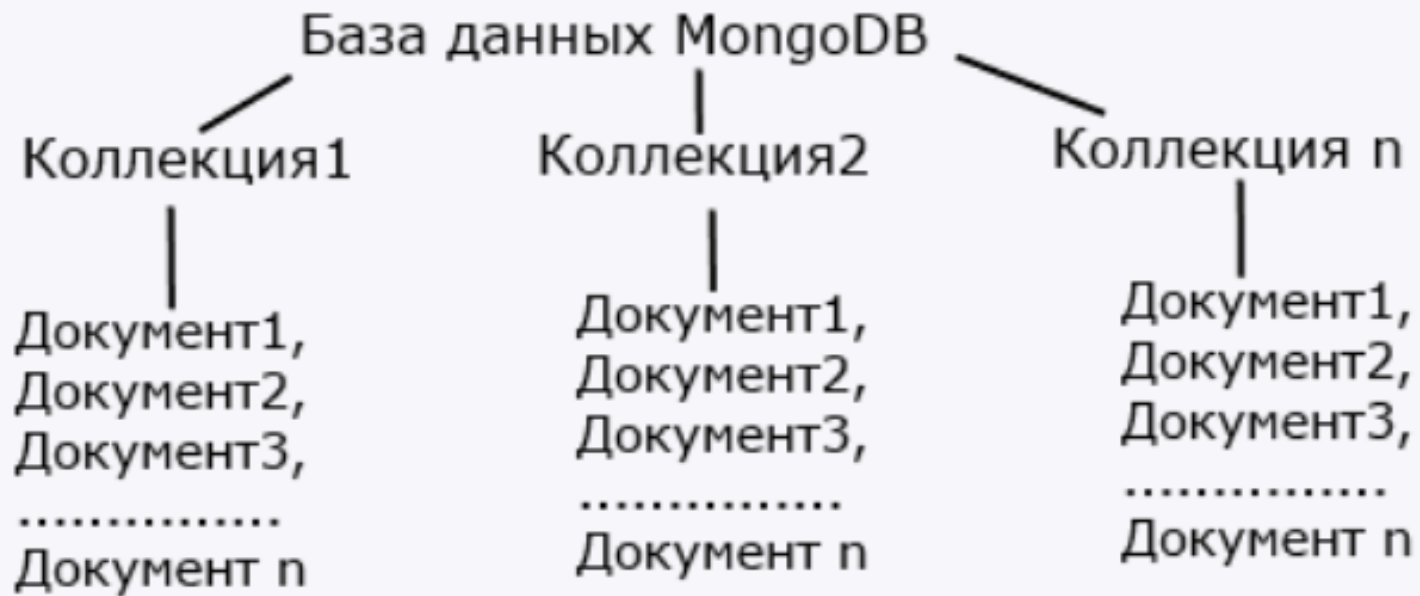
Attribute

Name
Price

Value

Simple Values
Linked to attributes






```
1 {
2     "name": "Bill",
3     "surname": "Gates",
4     "age": "48",
5     "company": {
6         "name" : "microsoft",
7         "year" : "1974",
8         "price" : "300000"
9     }
10 }
```

Для добавления в коллекцию могут использоваться три ее метода:

- **insertOne()**: добавляет один документ
- **insertMany()**: добавляет несколько документов
- **insert()**: может добавлять как один, так и несколько документов

```
> db.users.insertOne({"name": "Tom", "age": 28, languages: ["english", "spanish"]})
```

```
> db.users.insertMany([{"name": "Bob", "age": 26, languages: ["english", "frensh"]},  
{"name": "Alice", "age": 31, languages:["german", "english"]}])
```

```
> db.users.insert({"name": "Tom", "age": 28, languages: ["english", "spanish"]})
```

```
1 > db.users.find().pretty()
```

C:\mongodb\bin\mongo.exe

```
> db.users.find().pretty()
{
  "_id" : ObjectId("59382d2149f337809f210c67"),
  "name" : "Tom",
  "age" : 28,
  "languages" : [
    "english",
    "spanish"
  ]
}
{
  "_id" : ObjectId("59383270032ed06deaffacd7"),
  "name" : "Bob",
  "age" : 26,
  "languages" : [
    "english",
```

```
> db.users.insertOne({"name": "Tom", "age": 28, languages: ["english", "spanish"]})
> db.users.insertOne({"name": "Bill", "age": 32, languages: ["english", "french"]})
> db.users.insertOne({"name": "Tom", "age": 32, languages: ["english", "german"]})
```

```
> db.users.find({name: "Tom"})
```

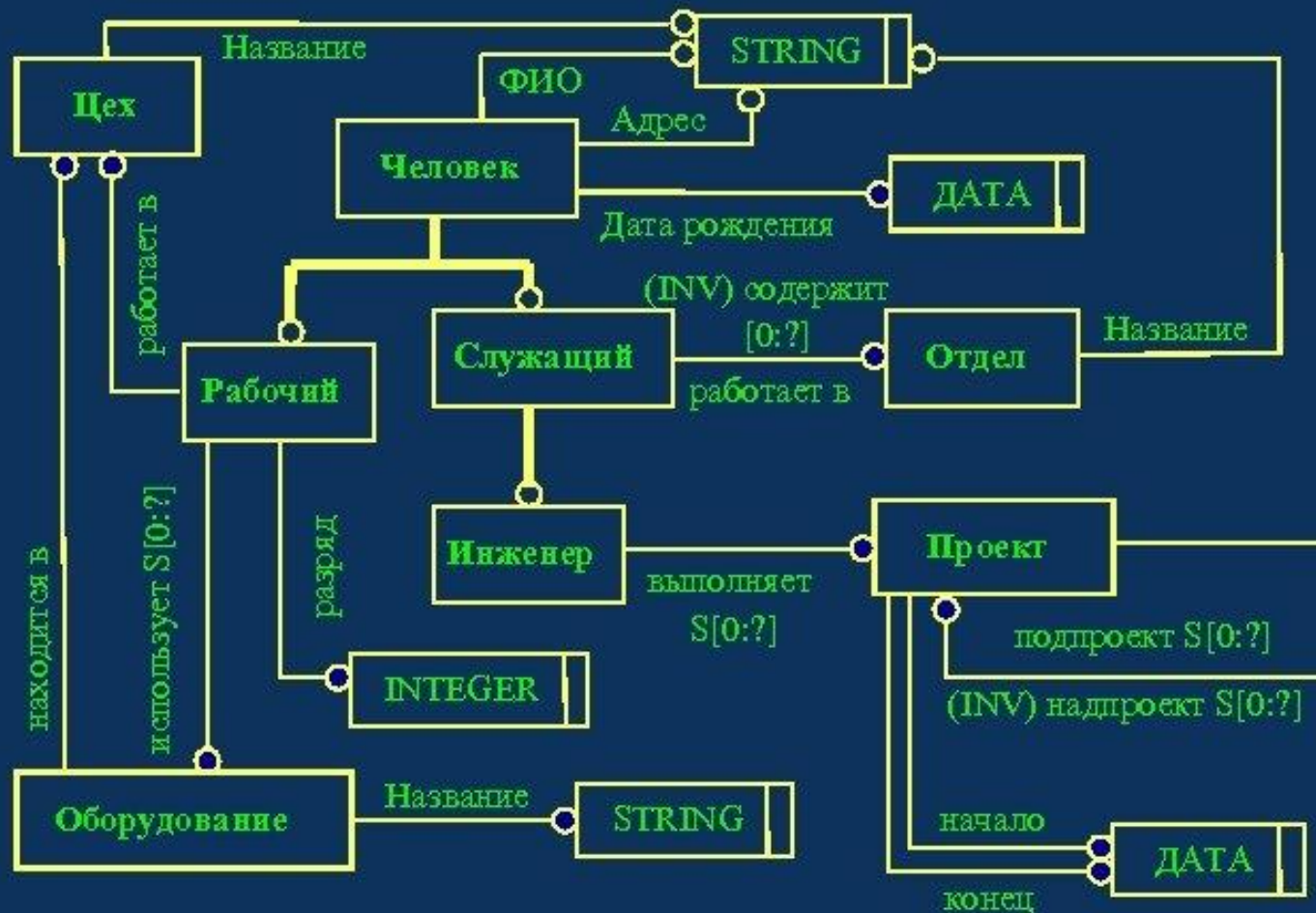
```
> db.users.find({name: "Tom", age: 32})
```

```
> db.users.find({languages: ["english", "german"]})
```

```
> db.users.find({languages: ["english", "german"]})
```

```
> db.users.find({languages.0: "english"})
```

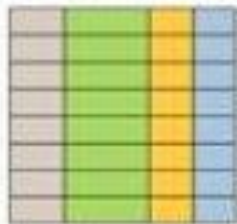
Пример схемы данных на языке EXPRESS-G



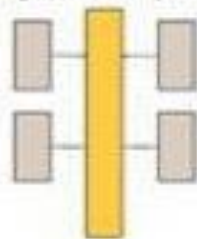
3. Программное обеспечение автоматизированных систем управления жизненным циклом изделия

SQL Databases

Relational

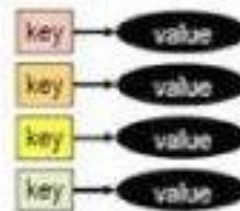


Analytical (OLAP)

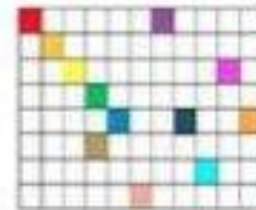


Non-SQL Databases

Key-Value



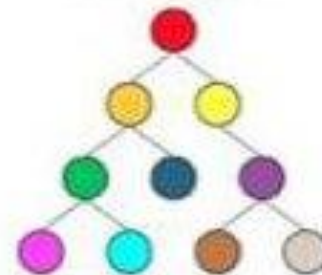
Column-Family

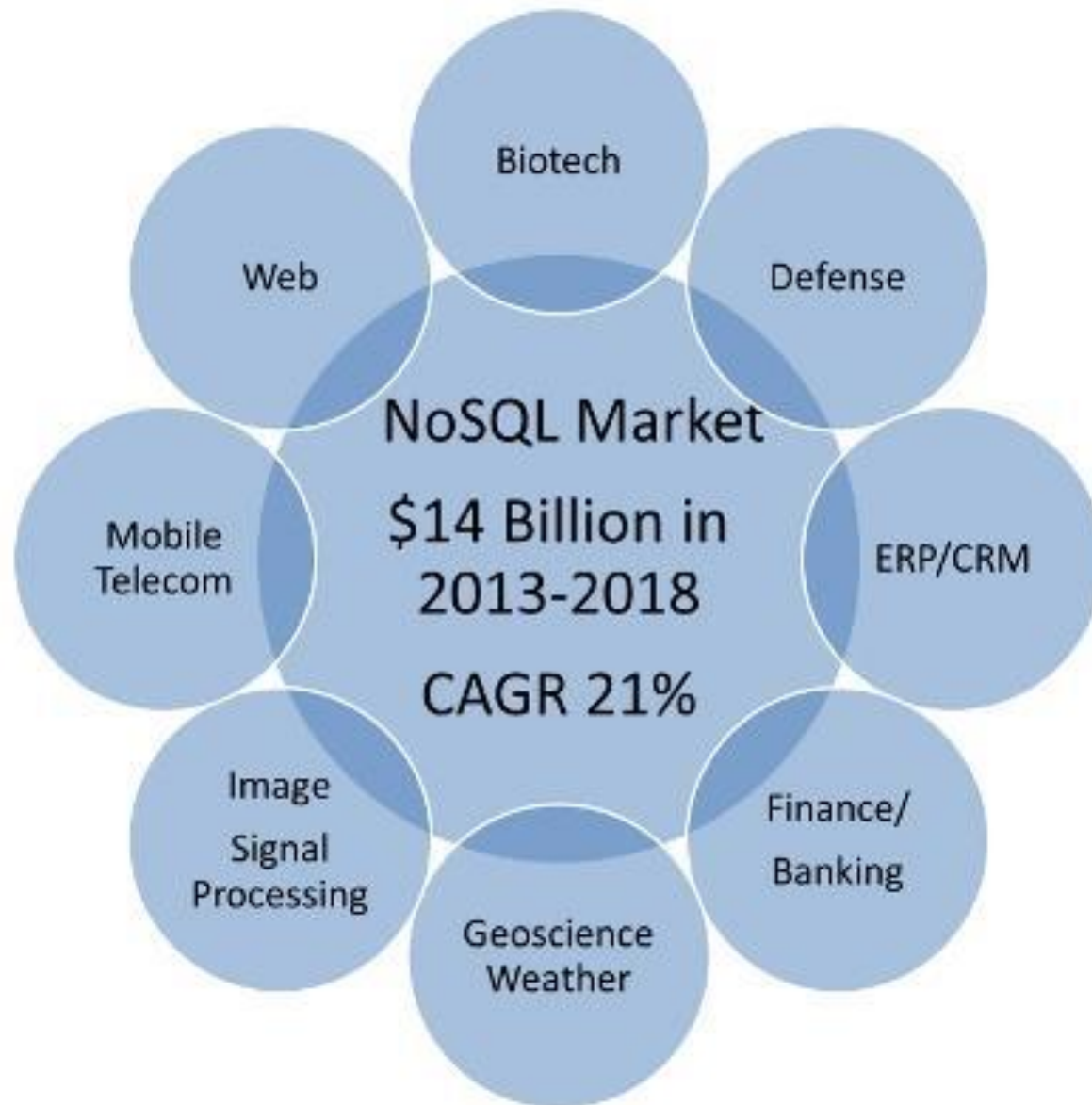


Graph



Document





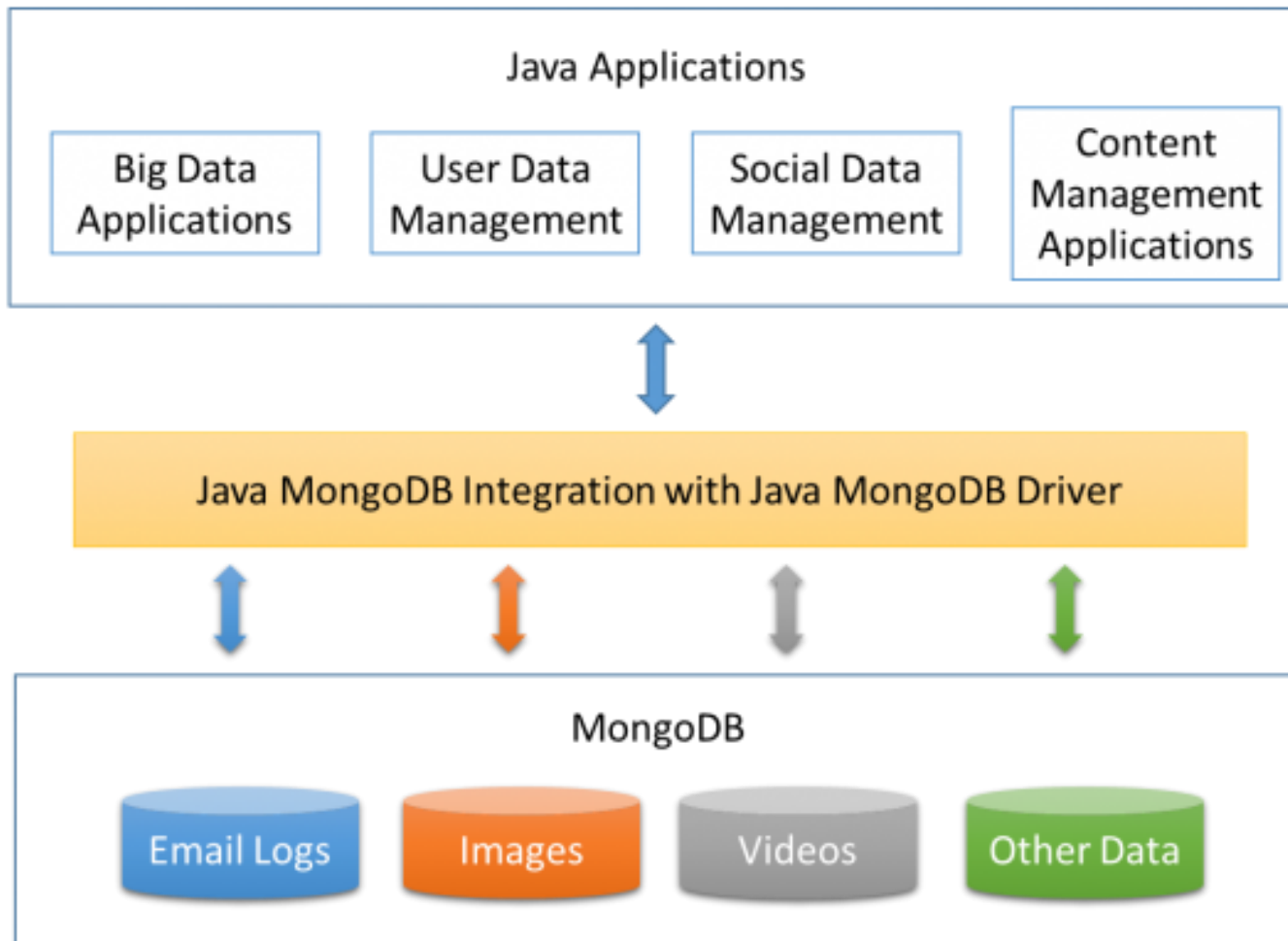
mySQL

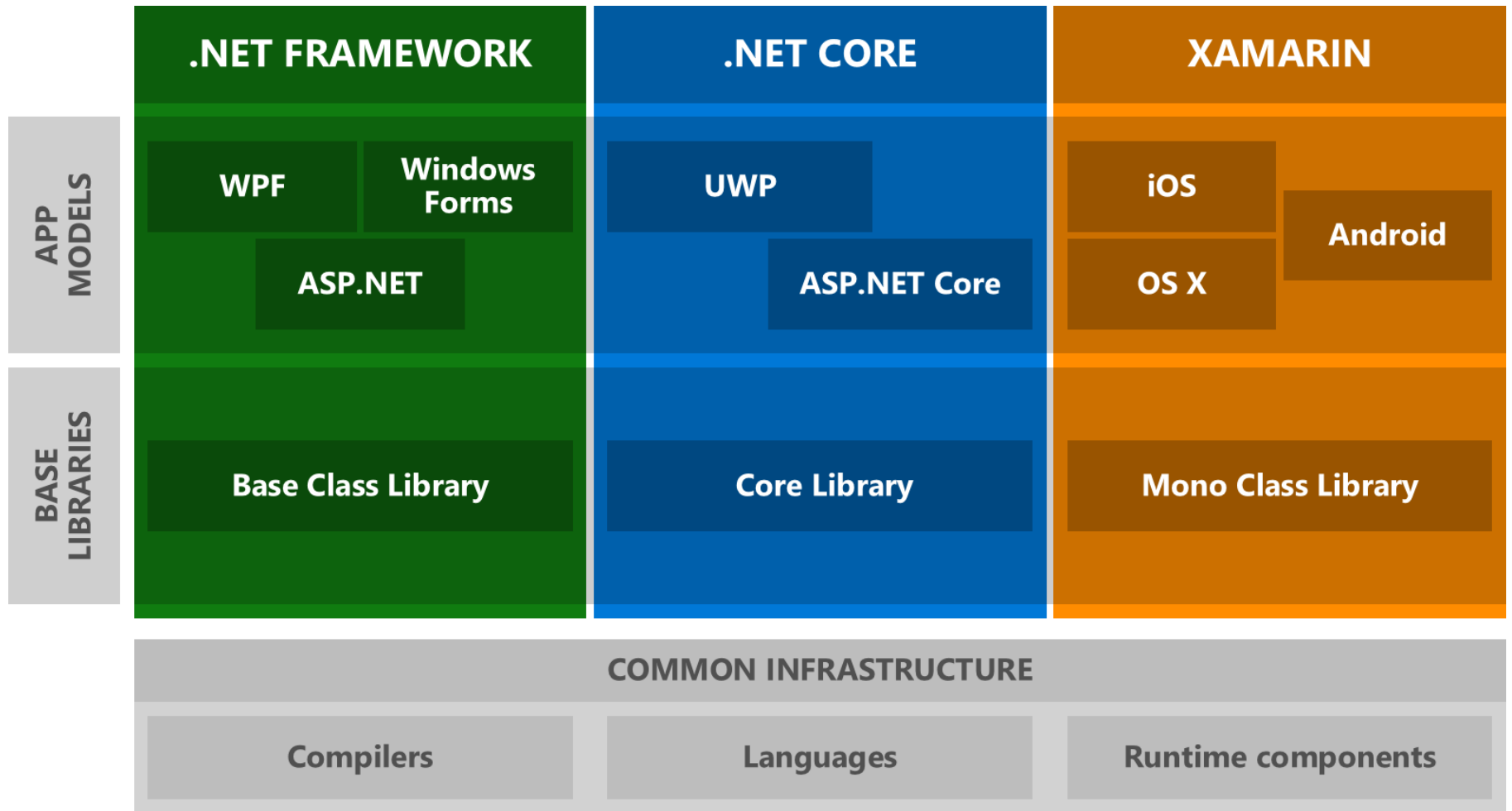
MongoDB

```
SELECT
  Dim1, Dim2,
  SUM(Measure1) AS MSum,
  COUNT(*) AS RecordCount,
  AVG(Measure2) AS MAvg,
  MIN(Measure1) AS MMin
  MAX(CASE
    WHEN Measure2 < 100
    THEN Measure2
  END) AS MMax
FROM DenormAggTable
WHERE (Filter1 IN ('A','B'))
  AND (Filter2 = 'C')
  AND (Filter3 > 123)
GROUP BY Dim1, Dim2
HAVING (MMin > 0)
ORDER BY RecordCount DESC
LIMIT 4, 8
```

```
db.runCommand({
  mapreduce: "DenormAggCollection",
  query: {
    filter1: { '$in': [ 'A', 'B' ] },
    filter2: 'C',
    filter3: { '$gt': 123 }
  },
  map: function() { emit(
    { d1: this.Dim1, d2: this.Dim2 },
    { msum: this.measure1, recs: 1, mmin: this.measure1,
      mmax: this.measure2 < 100 ? this.measure2 : 0 }
  );},
  reduce: function(key, vals) {
    var ret = { msum: 0, recs: 0, mmin: 0, mmax: 0 };
    for(var i = 0; i < vals.length; i++) {
      ret.msum += vals[i].msum;
      ret.recs += vals[i].recs;
      if(vals[i].mmin < ret.mmin) ret.mmin = vals[i].mmin;
      if((vals[i].mmax < 100) && (vals[i].mmax > ret.mmax))
        ret.mmax = vals[i].mmax;
    }
    return ret;
  },
  finalize: function(key, val) {
    val.mavg = val.msum / val.recs;
    return val;
  },
  out: 'result1',
  verbose: true
});
db.result1.
  find({ mmin: { '$gt': 0 } }).
  sort({ recs: -1 }).
  skip(4).
  limit(8);
```

- ① Grouped dimension columns are pulled out as keys in the map function, reducing the size of the working set.
- ② Measures must be manually aggregated.
- ③ Aggregates depending on record counts must wait until finalization.
- ④ Measures can use procedural logic.
- ⑤ Filters have an ORM/ActiveRecord-looking style.
- ⑥ Aggregate filtering must be applied to the result set, not in the map/reduce.
- ⑦ Ascending: 1; Descending: -1





.NET FRAMEWORK

.NET CORE

XAMARIN

APP MODELS

WPF

Windows Forms

UWP

ASP.NET

ASP.NET Core

iOS

Android

OS X

BASE LIBRARIES

Base Class Library

Core Library

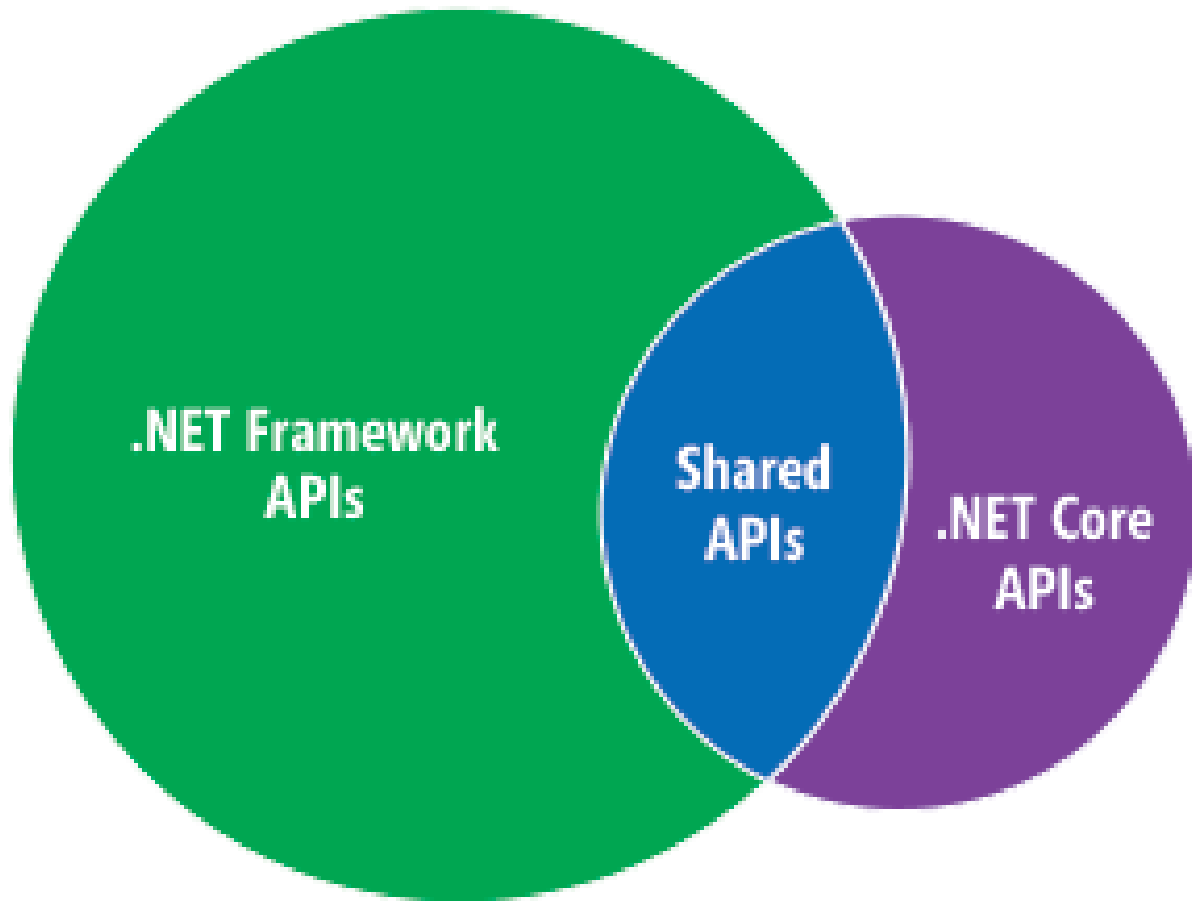
Mono Class Library

COMMON INFRASTRUCTURE

Compilers

Languages

Runtime components

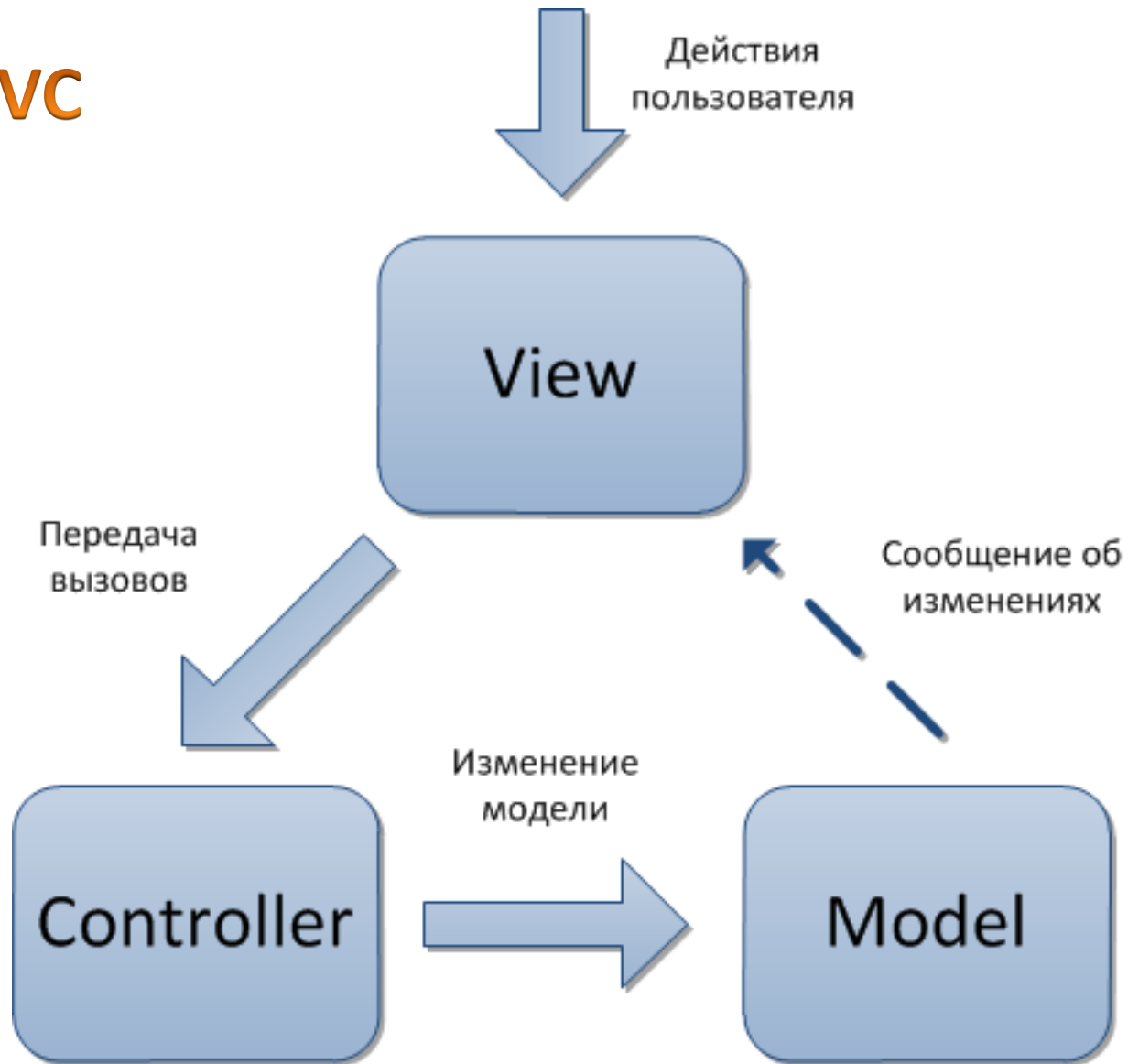


**.NET Framework
APIs**

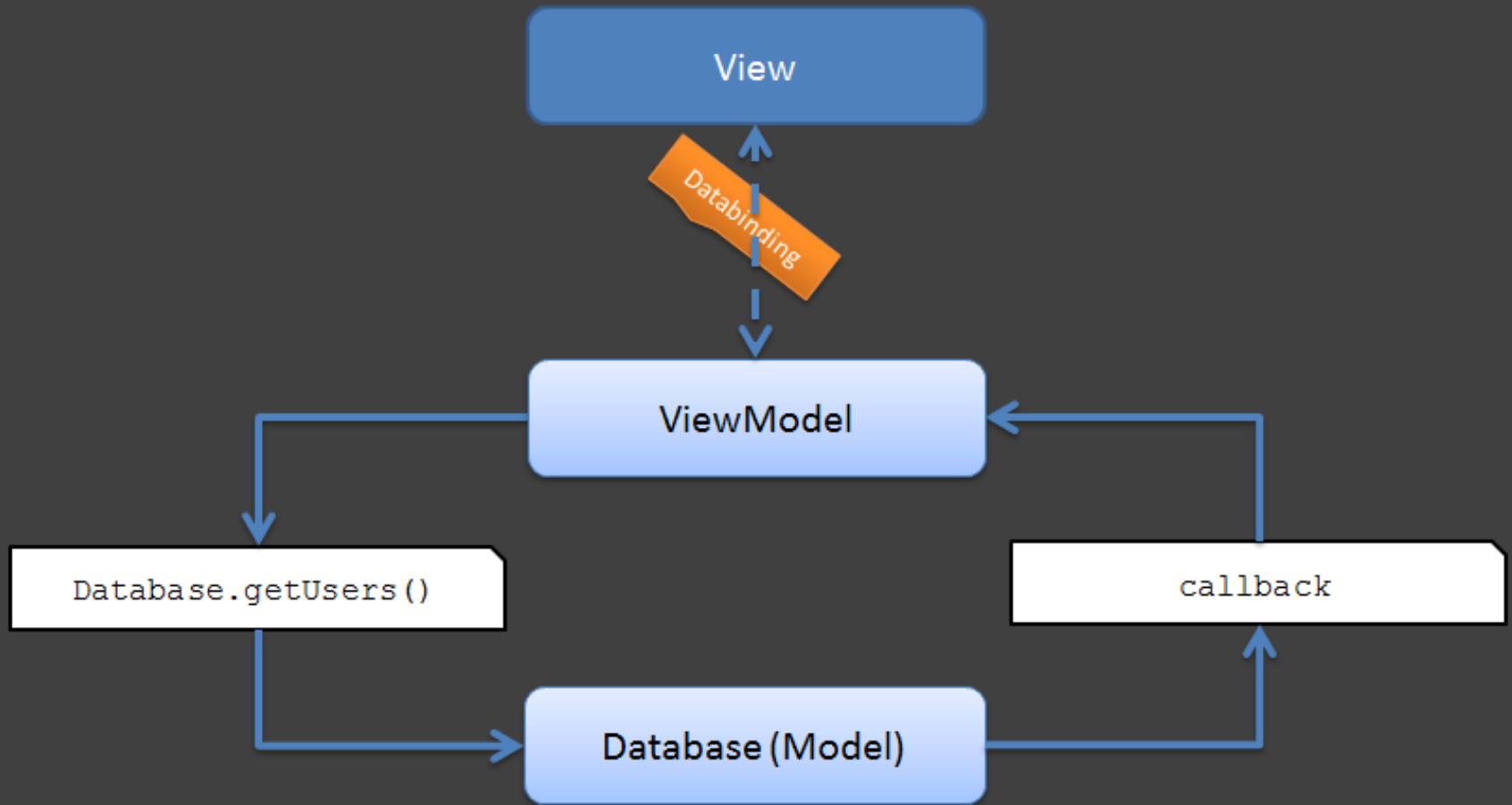
**Shared
APIs**

**.NET Core
APIs**

MVC



MVVM



C#

```
public class Startup
{
    public void ConfigureServices(IServiceCollection services)
    {
        // Includes support for Razor Pages and controllers.
        services.AddMvc();
    }

    public void Configure(IApplicationBuilder app)
    {
        app.UseMvc();
    }
}
```

```
@page
@using RazorPages
@model IndexModel2
```

```
<h2>Separate page model</h2>
<p>
    @Model.Message
</p>
```

```
using Microsoft.AspNetCore.Mvc.RazorPages;
using System;

namespace RazorPages
{
    public class IndexModel2 : PageModel
    {
        public string Message { get; private set; } = "PageModel in C#";

        public void OnGet()
        {
            Message += $" Server time is { DateTime.Now }";
        }
    }
}
```

/Pages/Index.cshtml

`/` or `/Index`

/Pages/Contact.cshtml

`/Contact`

/Pages/Store/Contact.cshtml

`/Store/Contact`

/Pages/Store/Index.cshtml

`/Store` or `/Store/Index`

```
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Hosting;
using Microsoft.EntityFrameworkCore;
using Microsoft.Extensions.DependencyInjection;
using RazorPagesContacts.Data;

namespace RazorPagesContacts
{
    public class Startup
    {
        public IHostingEnvironment HostingEnvironment { get; }

        public void ConfigureServices(IServiceCollection services)
        {
            services.AddDbContext<AppDbContext>(options =>
                options.UseInMemoryDatabase("name"));
            services.AddMvc();
        }

        public void Configure(IApplicationBuilder app)
        {
            app.UseMvc();
        }
    }
}
```

```
using System.ComponentModel.DataAnnotations;

namespace RazorPagesContacts.Data
{
    public class Customer
    {
        public int Id { get; set; }

        [Required, StringLength(100)]
        public string Name { get; set; }
    }
}
```

```
using Microsoft.EntityFrameworkCore;

namespace RazorPagesContacts.Data
{
    public class AppDbContext : DbContext
    {
        public AppDbContext(DbContextOptions options)
            : base(options)
        {
        }

        public DbSet<Customer> Customers { get; set; }
    }
}
```

```
@page
```

```
@model RazorPagesContacts.Pages.CreateModel
```

```
@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers
```

```
<html>
```

```
<body>
```

```
  <p>
```

```
    Enter your name.
```

```
  </p>
```

```
  <div asp-validation-summary="All"></div>
```

```
  <form method="POST">
```

```
    <div>Name: <input asp-for="Customer.Name" /></div>
```

```
    <input type="submit" />
```

```
  </form>
```

```
</body>
```

```
</html>
```

```
namespace RazorPagesContacts.Pages
{
    public class CreateModel : PageModel
    {
        private readonly AppDbContext _db;

        public CreateModel(AppDbContext db)
        {
            _db = db;
        }

        [BindProperty]
        public Customer Customer { get; set; }

        public async Task<IActionResult> OnPostAsync()
        {
            if (!ModelState.IsValid)
            {
                return Page();
            }

            _db.Customers.Add(Customer);
            await _db.SaveChangesAsync();
            return RedirectToPage("/Index");
        }
    }
}
```

```
@page
@model RazorPagesContacts.Pages.IndexModel
@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers

<h1>Contacts</h1>
<form method="post">
  <table class="table">
    <thead>
      <tr>
        <th>ID</th>
        <th>Name</th>
      </tr>
    </thead>
    <tbody>
      @foreach (var contact in Model.Customers)
      {
        <tr>
          <td>@contact.Id</td>
          <td>@contact.Name</td>
          <td>
            <a asp-page="./Edit" asp-route-id="@contact.Id">edit</a>
            <button type="submit" asp-page-handler="delete"
              asp-route-id="@contact.Id">delete</button>
          </td>
        </tr>
      }
    </tbody>
  </table>

  <a asp-page="./Create">Create</a>
</form>
```

```
namespace RazorPagesContacts.Pages
{
    public class IndexModel : PageModel
    {
        private readonly AppDbContext _db;

        public IndexModel(AppDbContext db)
        {
            _db = db;
        }

        public IList<Customer> Customers { get; private set; }

        public async Task OnGetAsync()
        {
            Customers = await _db.Customers.AsNoTracking().ToListAsync();
        }

        public async Task<IActionResult> OnPostDeleteAsync(int id)
        {
            var contact = await _db.Customers.FindAsync(id);

            if (contact != null)
            {
                _db.Customers.Remove(contact);
                await _db.SaveChangesAsync();
            }

            return RedirectToPage();
        }
    }
}
```

```
@page "{id:int}"
@model RazorPagesContacts.Pages.EditModel
@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers

@{
    ViewData["Title"] = "Edit Customer";
}

<h1>Edit Customer - @Model.Customer.Id</h1>
<form method="post">
    <div asp-validation-summary="All"></div>
    <input asp-for="Customer.Id" type="hidden" />
    <div>
        <label asp-for="Customer.Name"></label>
        <div>
            <input asp-for="Customer.Name" />
            <span asp-validation-for="Customer.Name" ></span>
        </div>
    </div>

    <div>
        <button type="submit">Save</button>
    </div>
</form>
```



```
namespace RazorPagesContacts.Pages
{
    public class EditModel : PageModel
    {
        private readonly AppDbContext _db;

        public EditModel(AppDbContext db)
        {
            _db = db;
        }

        [BindProperty]
        public Customer Customer { get; set; }

        public async Task<IActionResult> OnGetAsync(int id)
        {
            Customer = await _db.Customers.FindAsync(id);

            if (Customer == null)
            {
                return RedirectToPage("/Index");
            }

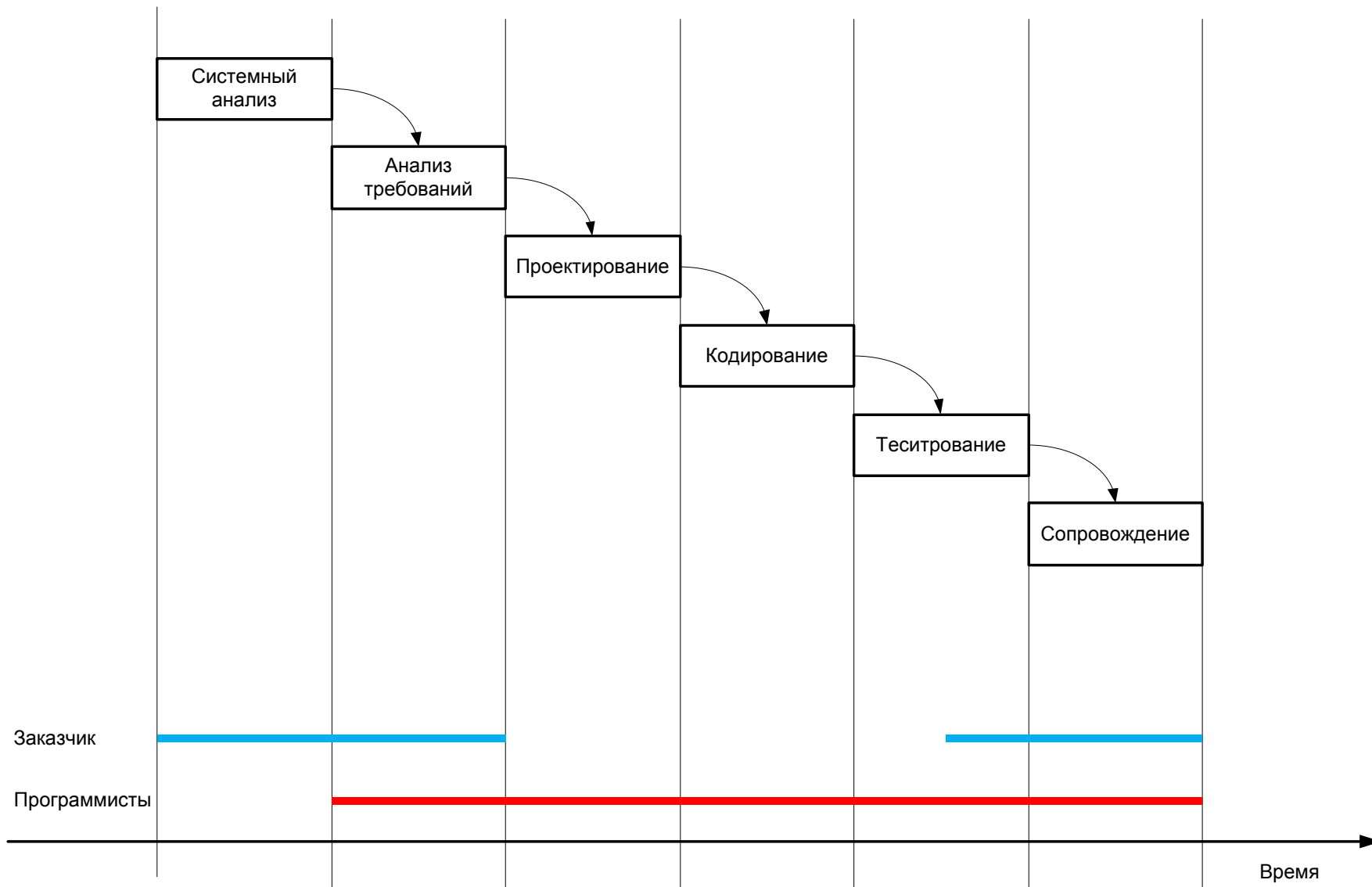
            return Page();
        }
    }
}
```

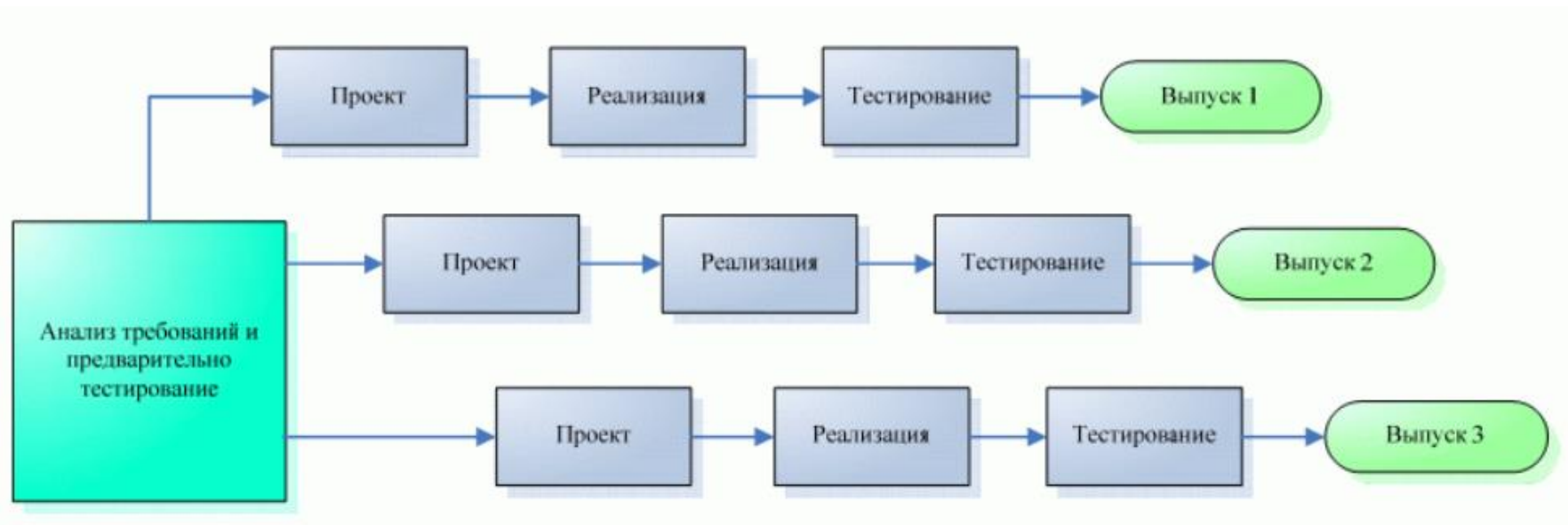
```
public async Task<IActionResult> OnPostAsync()
{
    if (!ModelState.IsValid)
    {
        return Page();
    }

    _db.Attach(Customer).State = EntityState.Modified;

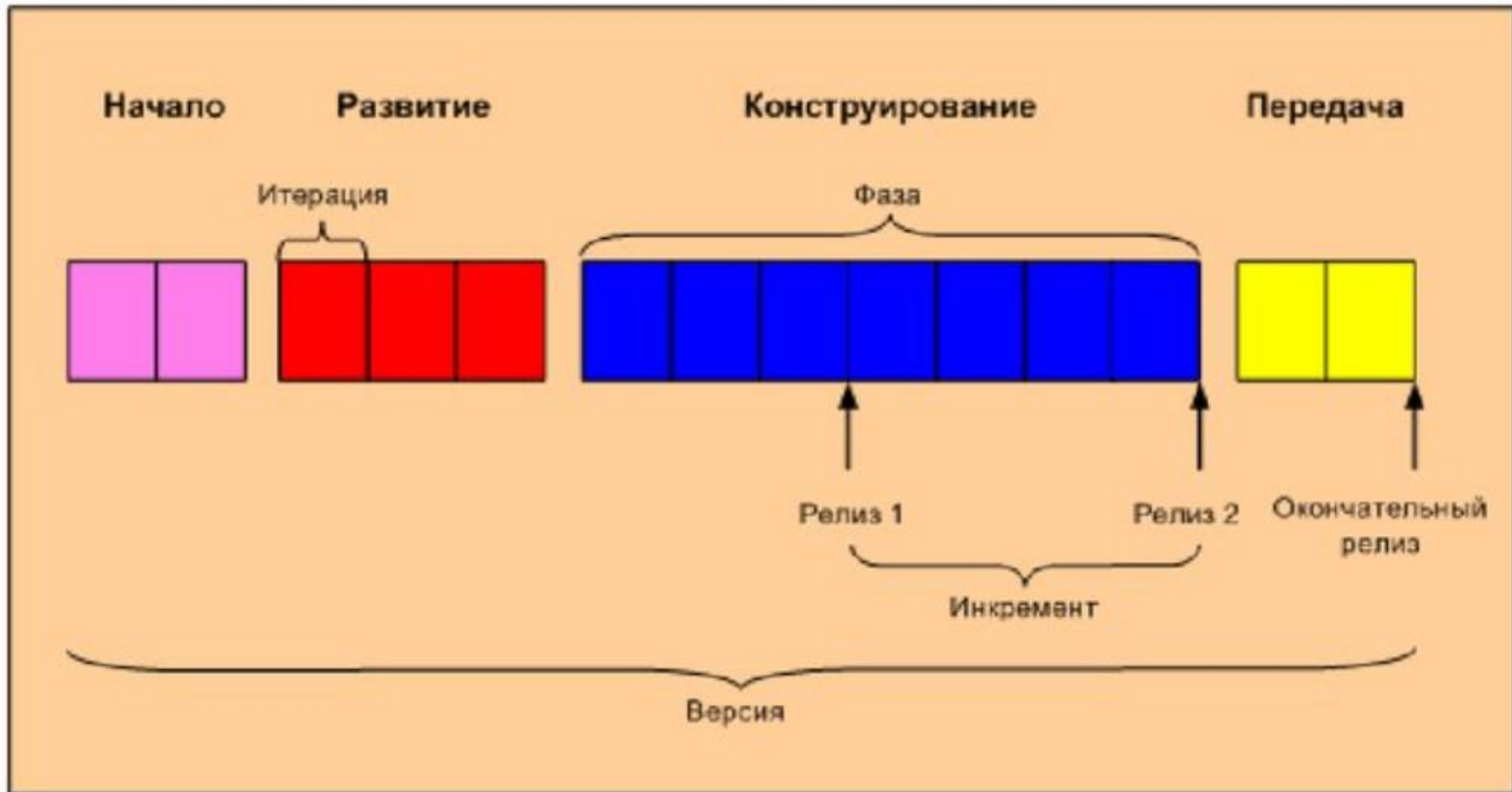
    try
    {
        await _db.SaveChangesAsync();
    }
    catch (DbUpdateConcurrencyException)
    {
        throw new Exception($"Customer {Customer.Id} not found!");
    }

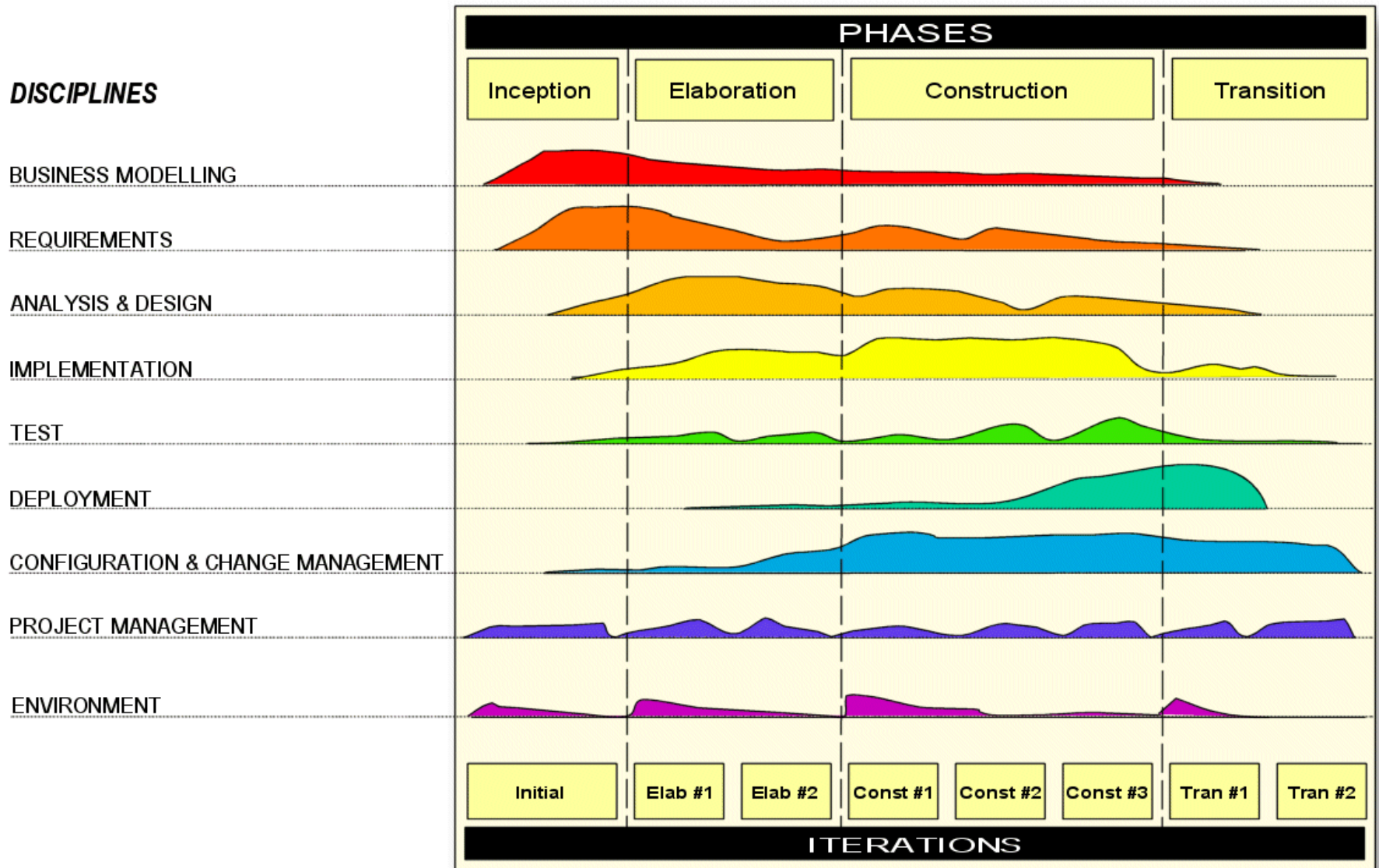
    return RedirectToPage("/Index");
}
}
```





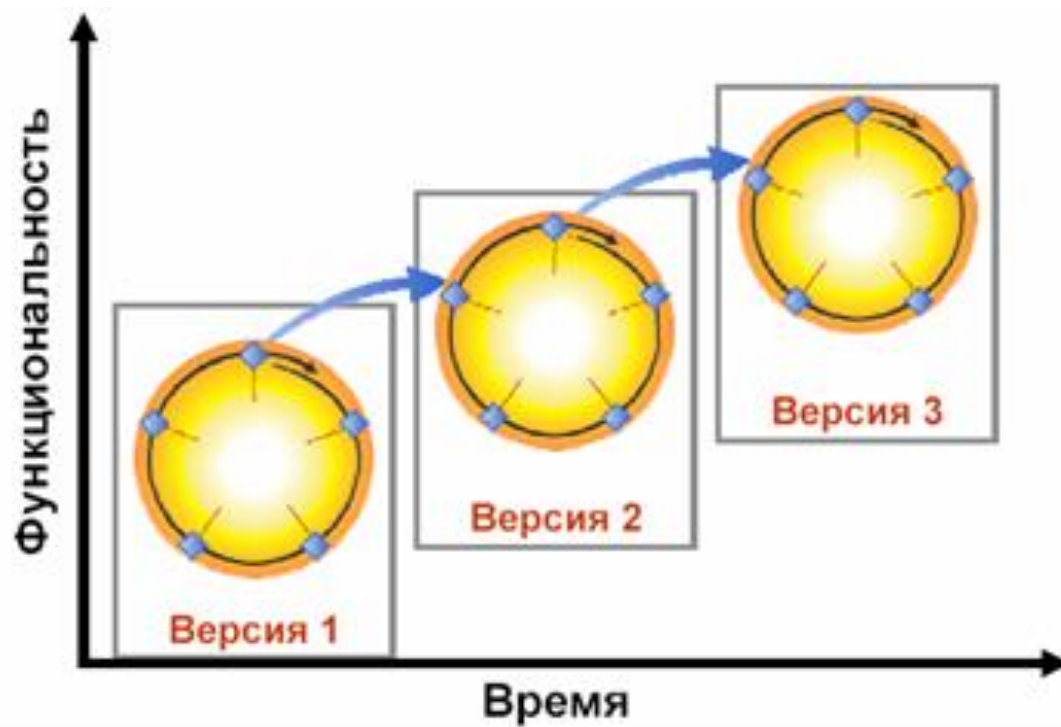






Microsoft Solutions Framework

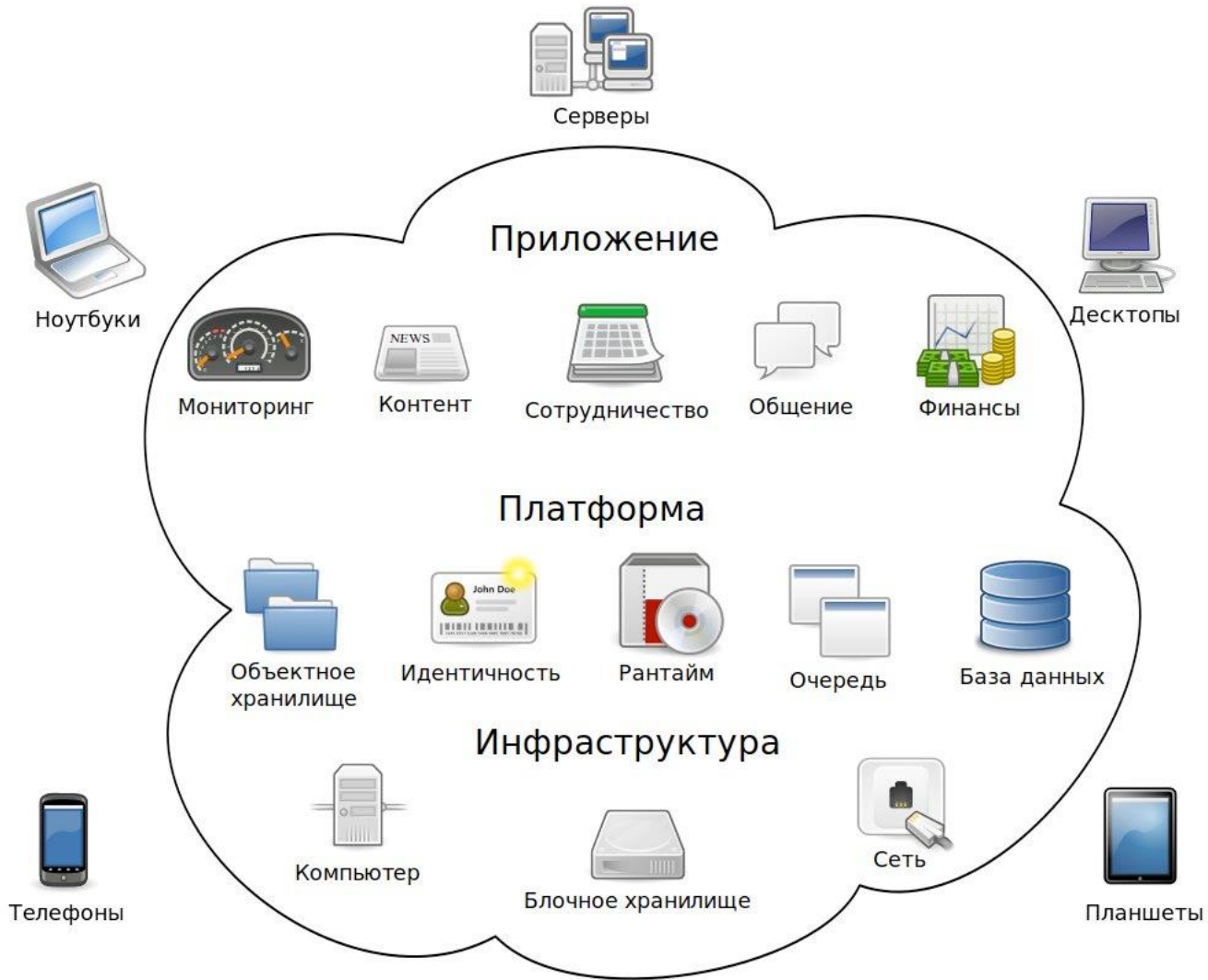




Стадии разработки	Этапы работ	Содержание работ
1. Техническое задание	Обоснование необходимости разработки программы	<p>Постановка задачи</p> <p>Сбор исходных материалов</p> <p>Выбор и обоснование критериев эффективности и качества разрабатываемой программы.</p> <p>Обоснование необходимости проведения научно-исследовательских работ.</p>
	Научно-исследовательские работы	<p>Определение структуры входных и выходных данных.</p> <p>Предварительный выбор методов решения задач.</p> <p>Обоснование целесообразности применения ранее разработанных программ.</p> <p>Определение требований к техническим средствам.</p> <p>Обоснование принципиальной возможности решения поставленной задачи</p>
	Разработка и утверждение технического задания	<p>Определение требований к программе.</p> <p>Разработка технико-экономического обоснования разработки программы.</p> <p>Определение стадий, этапов и сроков разработки программы и документации на неё.</p> <p>Выбор языков программирования.</p> <p>Определение необходимости проведения научно-исследовательских работ на последующих стадиях.</p> <p>Согласование и утверждение технического задания.</p>

2. Эскизный проект	Разработка эскизного проекта	<p>Предварительная разработка структуры входных и выходных данных.</p> <p>Уточнение методов решения задачи.</p> <p>Разработка общего описания алгоритма решения задачи</p> <p>Разработка технико-экономического обоснования.</p>
	Утверждение эскизного проекта	<p>Разработка пояснительной записки.</p> <p>Согласование и утверждение эскизного проекта.</p>
3. Технический проект	Разработка технического проекта	<p>Уточнение структуры входных и выходных данных.</p> <p>Разработка алгоритма решения задачи.</p> <p>Определение формы представления входных и выходных данных.</p> <p>Определение семантики и синтаксиса языка.</p> <p>Разработка структуры программы.</p> <p>Окончательное определение конфигурации технических средств.</p>
	Утверждение технического проекта	<p>Разработка плана мероприятий по разработке и внедрению программ.</p> <p>Разработка пояснительной записки.</p> <p>Согласование и утверждение технического проекта.</p>

4. Рабочий проект	Разработка программы	Программирование и отладка программы.
	Разработка программной документации	Разработка программных документов в соответствии с требованиями ГОСТ 19.101-77.
	Испытания программы	<p>Разработка, согласование и утверждение порядка и методики испытаний.</p> <p>Проведение предварительных государственных, межведомственных, приёмо-сдаточных и других видов испытаний.</p> <p>Корректировка программы и программной документации по результатам испытаний.</p>
5. Внедрение	Подготовка и передача программы.	<p>Подготовка и передача программы и программной документации для сопровождения и (или) изготовления.</p> <p>Оформление и утверждение акта о передаче программы на сопровождение и (или) изготовление.</p> <p>Передача программы в фонд алгоритмов и программ.</p>



ЧАСТНОЕ

ГИБРИДНОЕ

ОБЩЕДОСТУПНОЕ



КОНФИДЕНЦИАЛЬНЫЕ ДАННЫЕ

ВАЖНЫЕ РАБОЧИЕ НАГРУЗКИ

ЧАСТНОЕ

vmware

ГИБРИДНОЕ

ГИБКОСТЬ

Одна платформа

ОБЩЕДОСТУПНОЕ

vmware vCloud Air

2+ уровень

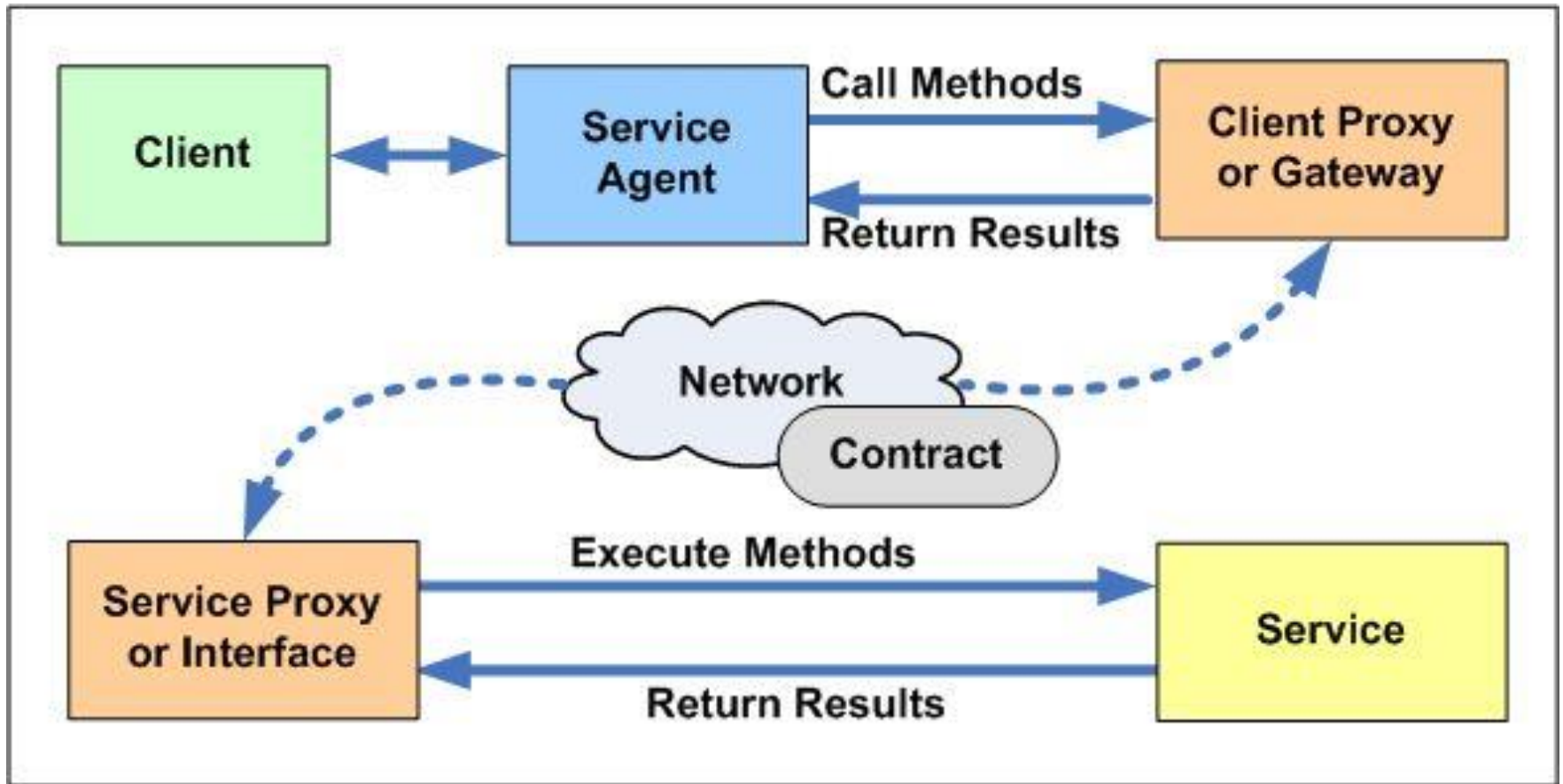
3+ уровень

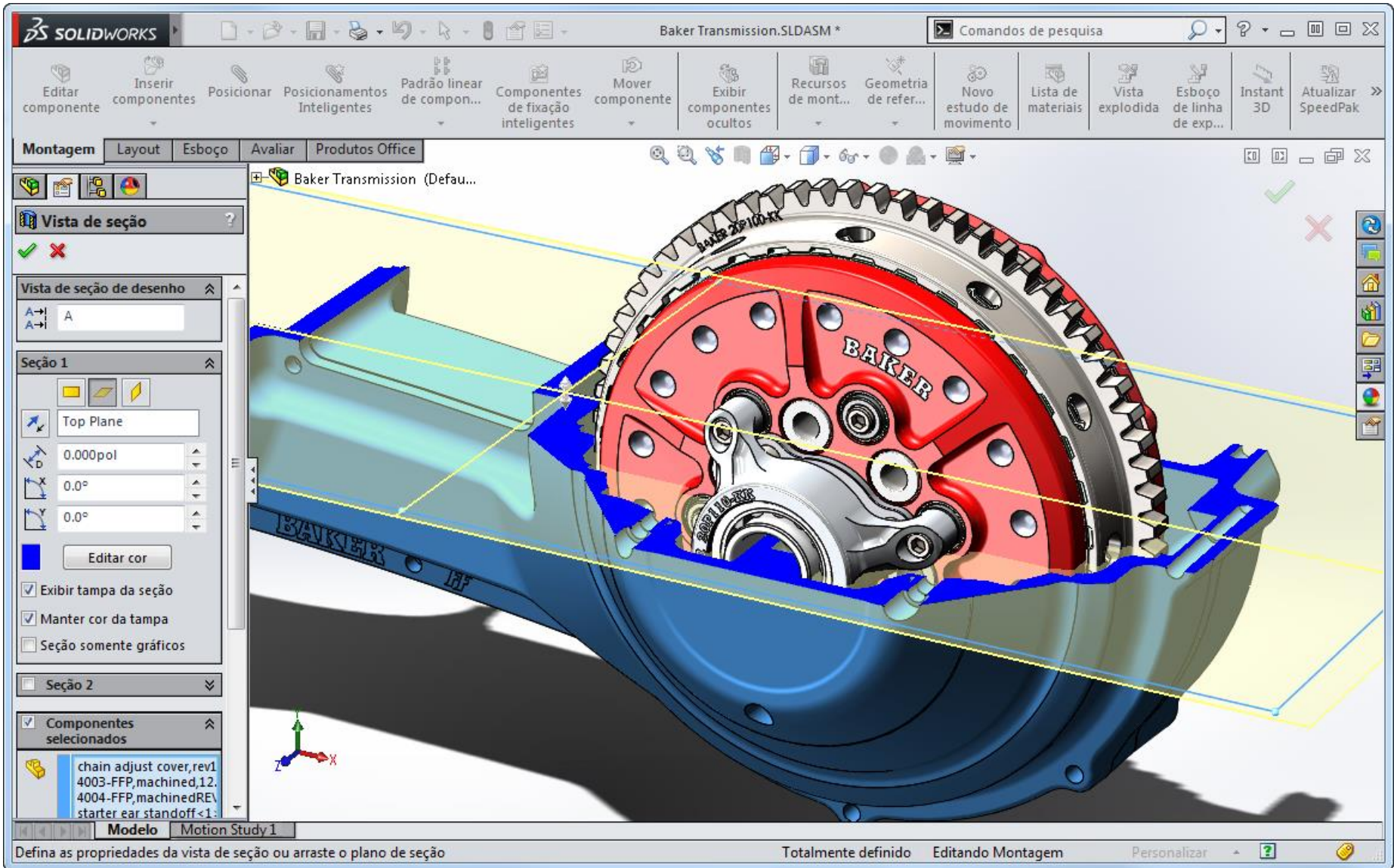


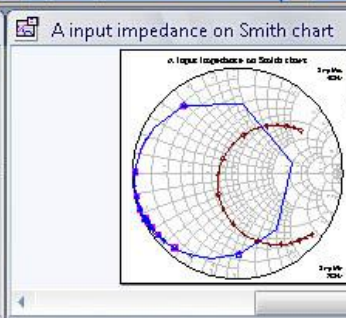
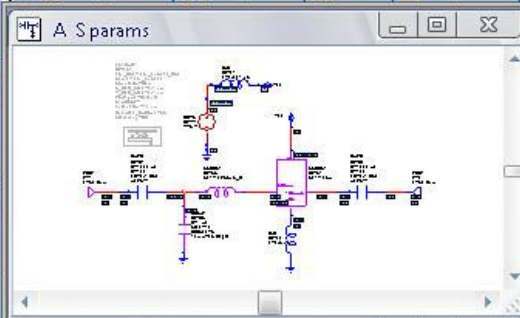
Безопасность частного облака
Экономичность общедоступного облака

Сокращение расходов









CMOS LNA with Variable Gain and Constant-Gm bias network

Wireless communication systems require low cost, low power and high performance. To achieve low cost, most RFICs are designed and fabricated in CMOS processes. The LNA shown in this example is all CMOS. Taking advantage of the complementary device process, we employ a constant-Gm bias network and an additional current branch in the amplifier topology to lower the gain by 6 dB with some additional CMOS logic. The additional branch is in parallel with the cascaded common-gate and will pull half the bias current of the main amplifier when activated. This ideally results in 6 dB lowering of the

