



SAPLAND

Современная отладка в АВАР. Через тернии к звёздам

Вебинар Никиты Калущкого

Москва, 21 октября 2021 г.

sapland.ru

- АВАР-разработчики
- Функциональные консультанты, разбирающиеся в АВАР
- Все, кому интересна автоматизация процессов разработки

- Познакомиться с инструментами, позволяющими значительно снизить трудоёмкость отладки.
- Повысить личную эффективность при разработке.
- Повысить качество кода.
- Экономить время на поддержку унаследованного кода.

- Раздел 1. Введение.
- Раздел 2. Отладка с учётом уровня.
- Раздел 3. Модульные (юнит) тесты.
- Раздел 4. Скрипты отладчика.

- Можно задавать вопросы голосом или в чате
- Интерактивное общение приветствуется
- Включение камеры приветствуется

Введение

*Человеку свойственно ошибаться, а
глупцу настаивать на ошибке.*

(Цицерон)

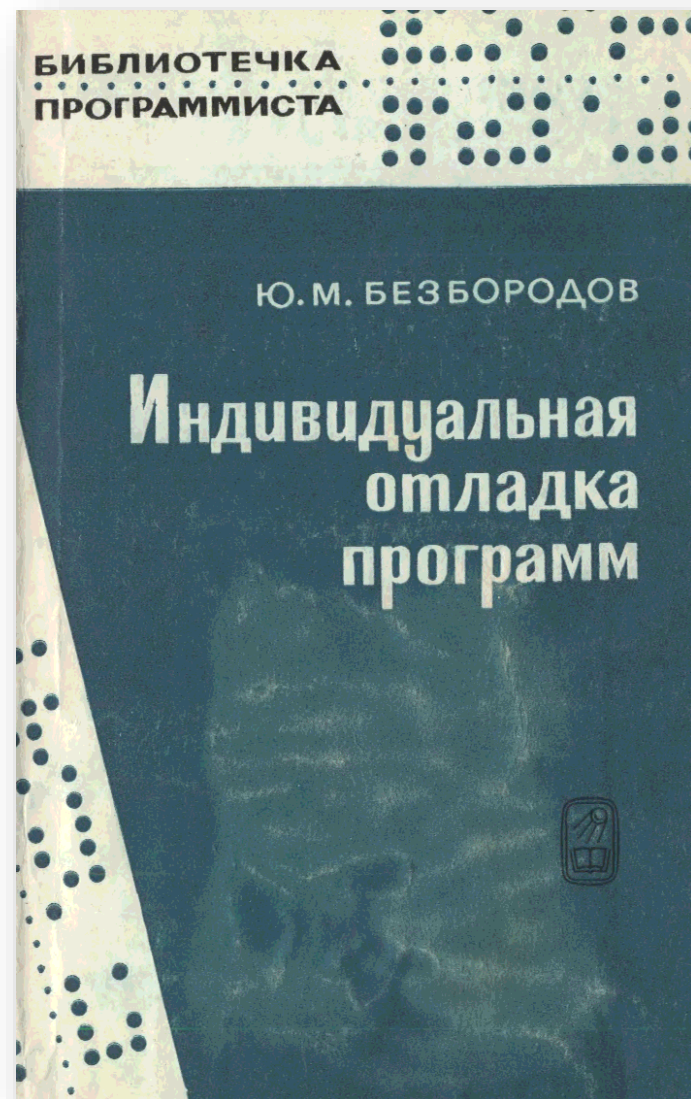
*В каждой программе есть по
крайней мере одна ошибка
(Из фольклора программистов)*

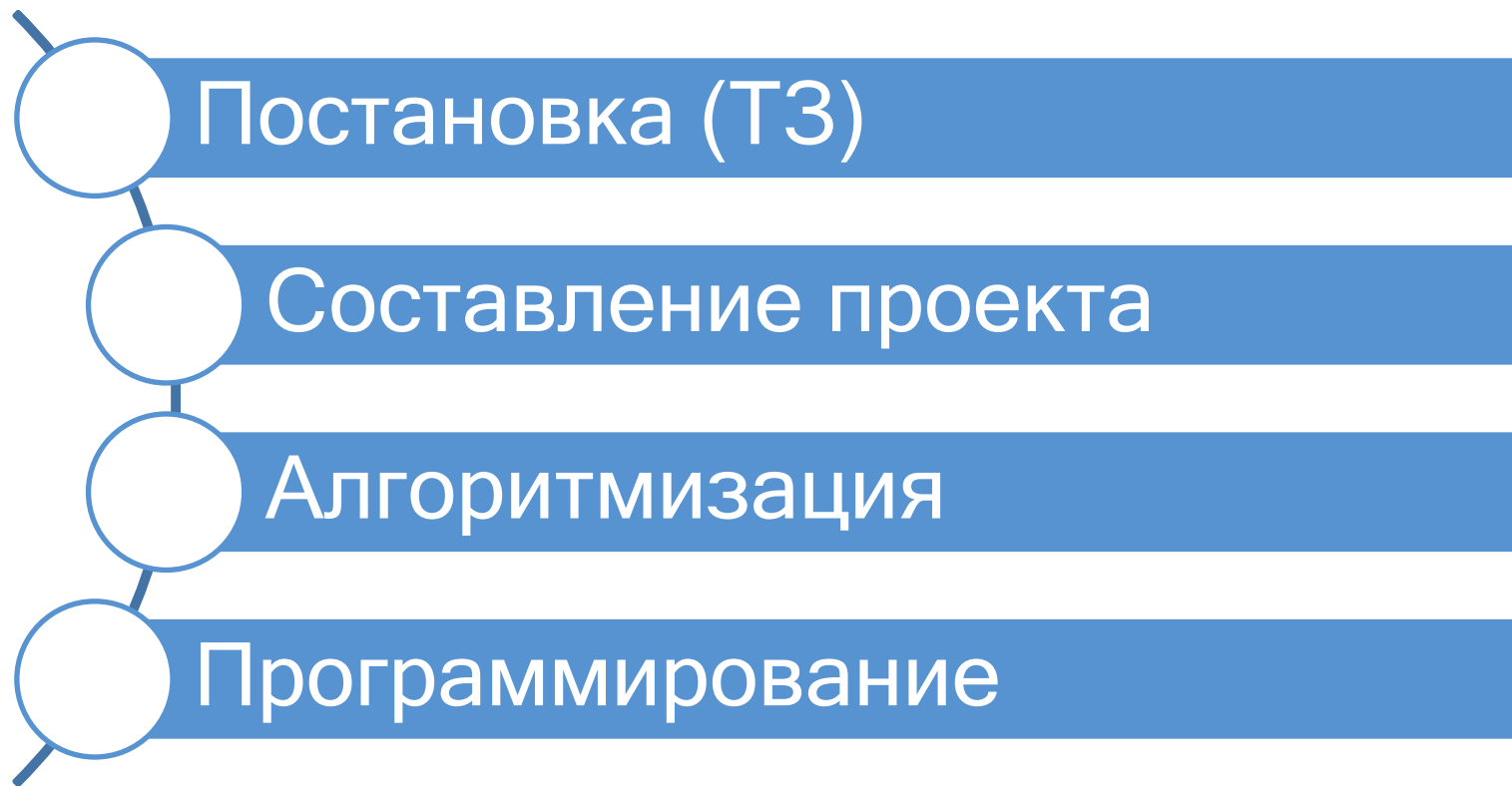


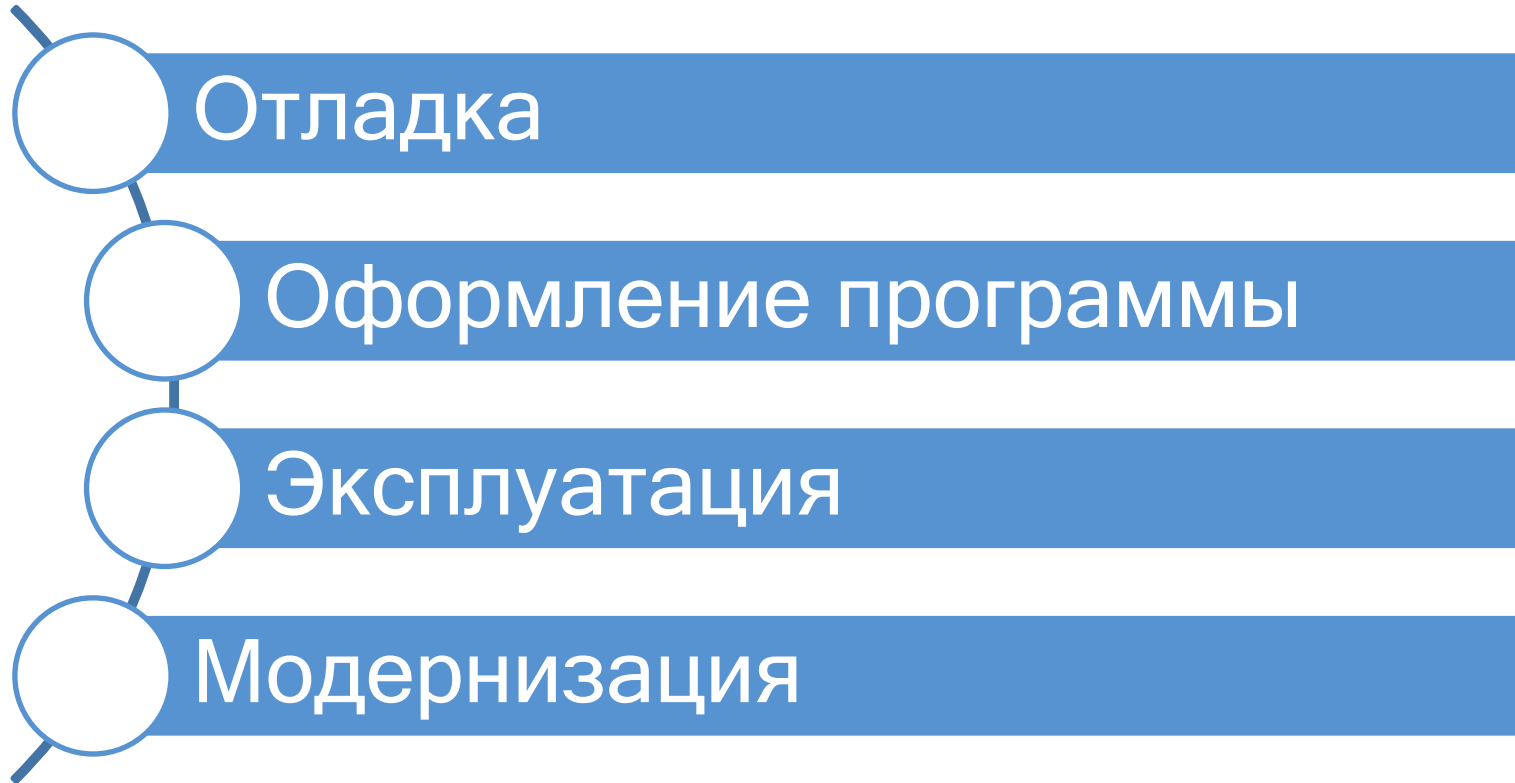


Ершов А.П. – выдающийся советский теоретик программирования

М.: Наука. Главная редакция физико-математической литературы, 1982





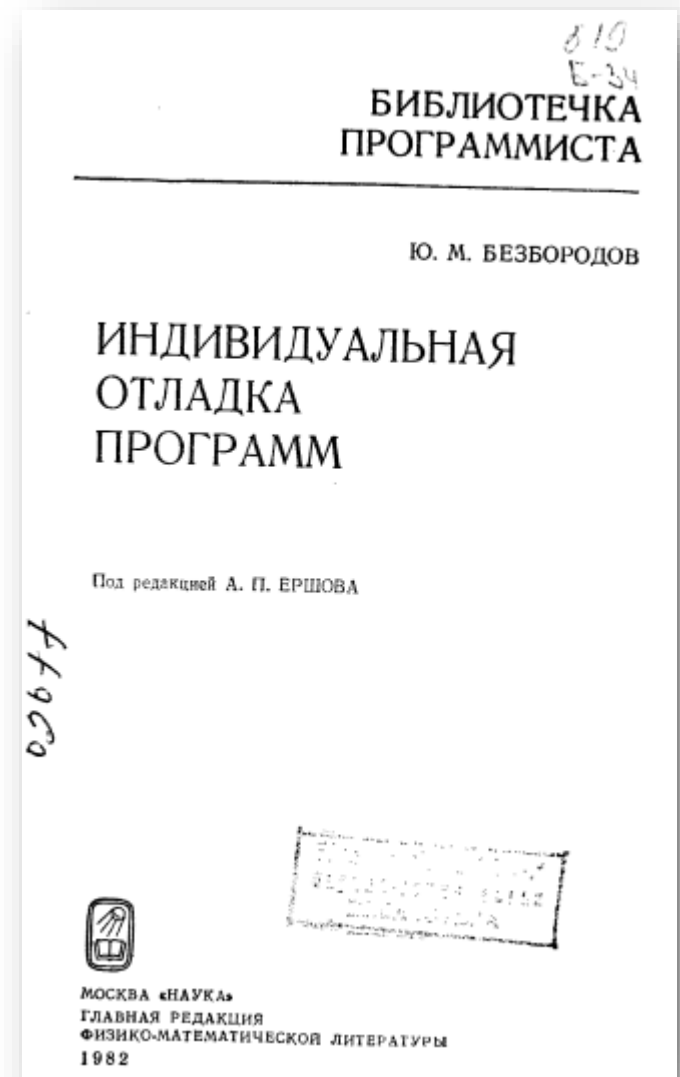


Отладка - этап разработки компьютерной программы, на котором обнаруживают, локализуют и устраняют ошибки.



Чтобы понять, где возникла ошибка, нужно:

- узнавать текущие значения переменных;
- выяснять, по какому пути выполнялась программа.



- Практически невозможно составить реальную программу без ошибок.
- Почти невозможно для достаточно сложной программы быстро найти и устранить все имеющиеся в ней ошибки.
- Следует признать ненормальным, из ряда вон выходящим фактом отсутствие ошибок в программе, которая не была ещё подвергнута тщательной отладке.



- Разумно уже при разработке программы готовиться к обнаружению ошибок на стадии отладки (юнит-тесты).
- При разработке алгоритма намечаются способы контроля отдельных блоков и приёмы предстоящей локализации ошибок в них.
- Чем более тщательно проведены этапы алгоритмизации и программирования и, в частности, чем более детально разработан план отладки, тем меньше времени потребуется на проведение самой отладки.



- Надежды на то, что устранение ошибок из программы произойдёт само собой никогда не оправдываются.
- Оптимизм и самоуверенность для программиста на стадии разработки противопоказаны.



Распределение трудоёмкости разработки ПО

№ этапа	Описание этапа	Трудоёмкость
1	Получение задания, составление проекта программы и общего плана отладки	10%
2	Разработка алгоритма и детального плана отладки	20%
3	Программирование и изготовление тестов	15%
4	Отладка	40%
5	Оформление программы	10%
6	Прочее	5%

Источники:

- 1) Брукс Э. Как проектируются и создаются программные комплексы. – М.: Наука, 1979.
- 2) Brown A.R., Sampson W.A. Program debugging. – New York: McDonald, 1973

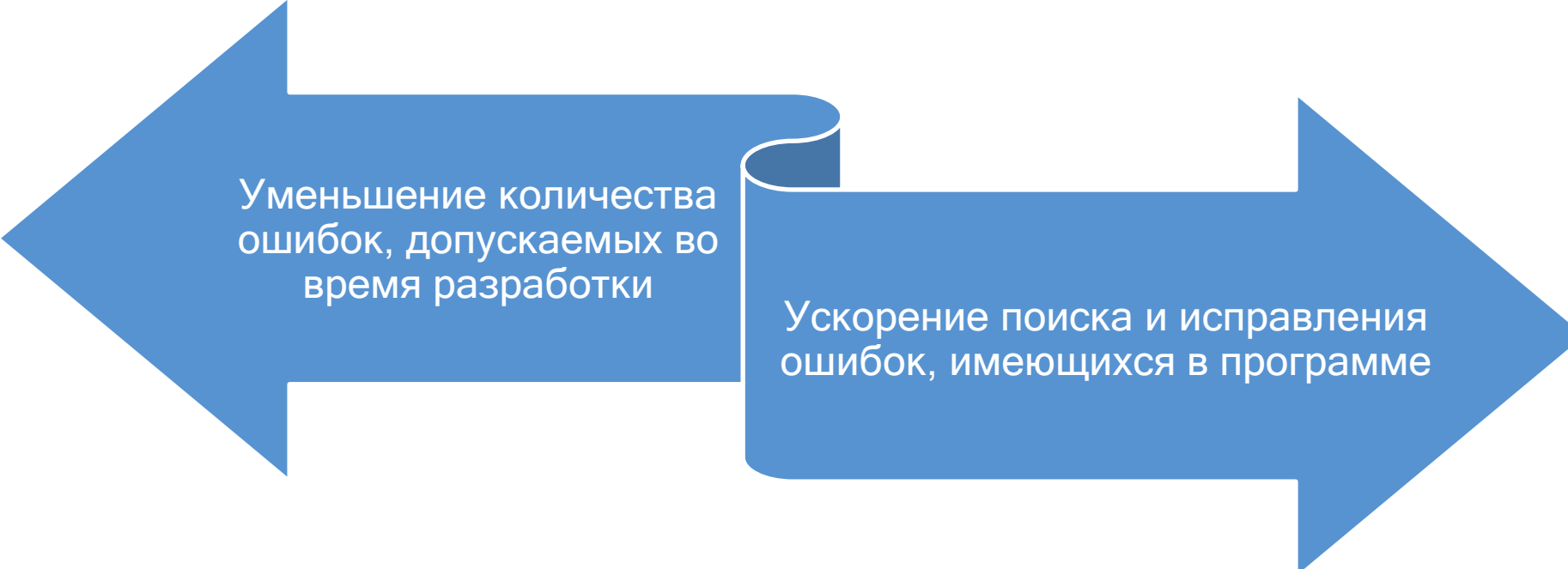
Распределение трудоёмкости разработки ПО

- Работы по доказательству (демонстрации) правильности разрабатываемой программы равнозначны работам по её изготовлению.

Разработка программы = изготовление + доказательство.

Время, затрачиваемое на работы, связанные с отладкой, составляет около половины всего времени, необходимого на разработку программы.

Что делать?

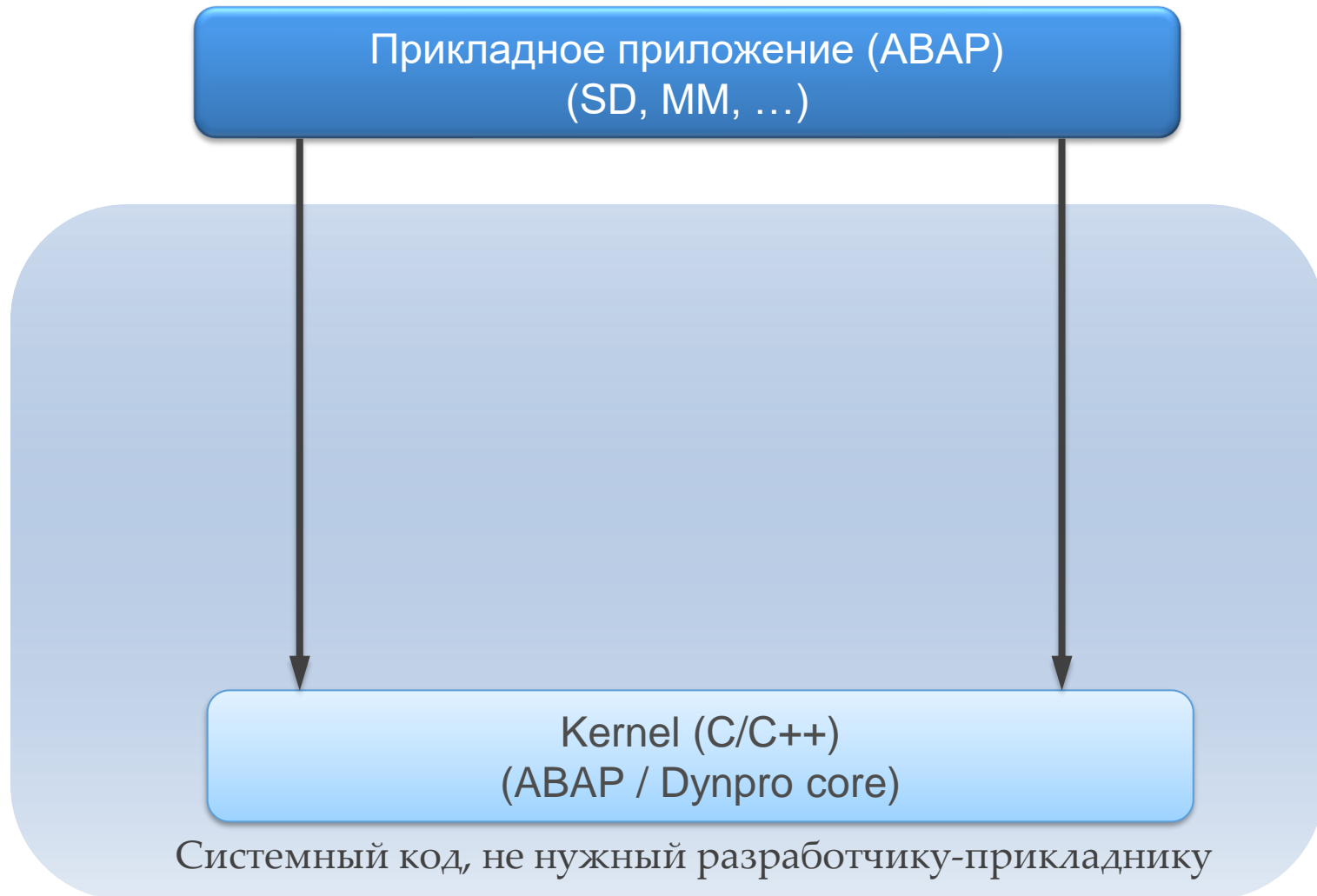
A diagram consisting of two large blue arrows pointing in opposite directions, one to the left and one to the right. They are connected at their inner ends by a curved, ribbon-like shape. The left arrow contains the text "Уменьшение количества ошибок, допускаемых во время разработки" and the right arrow contains the text "Ускорение поиска и исправления ошибок, имеющих в программе".

Уменьшение количества ошибок, допускаемых во время разработки

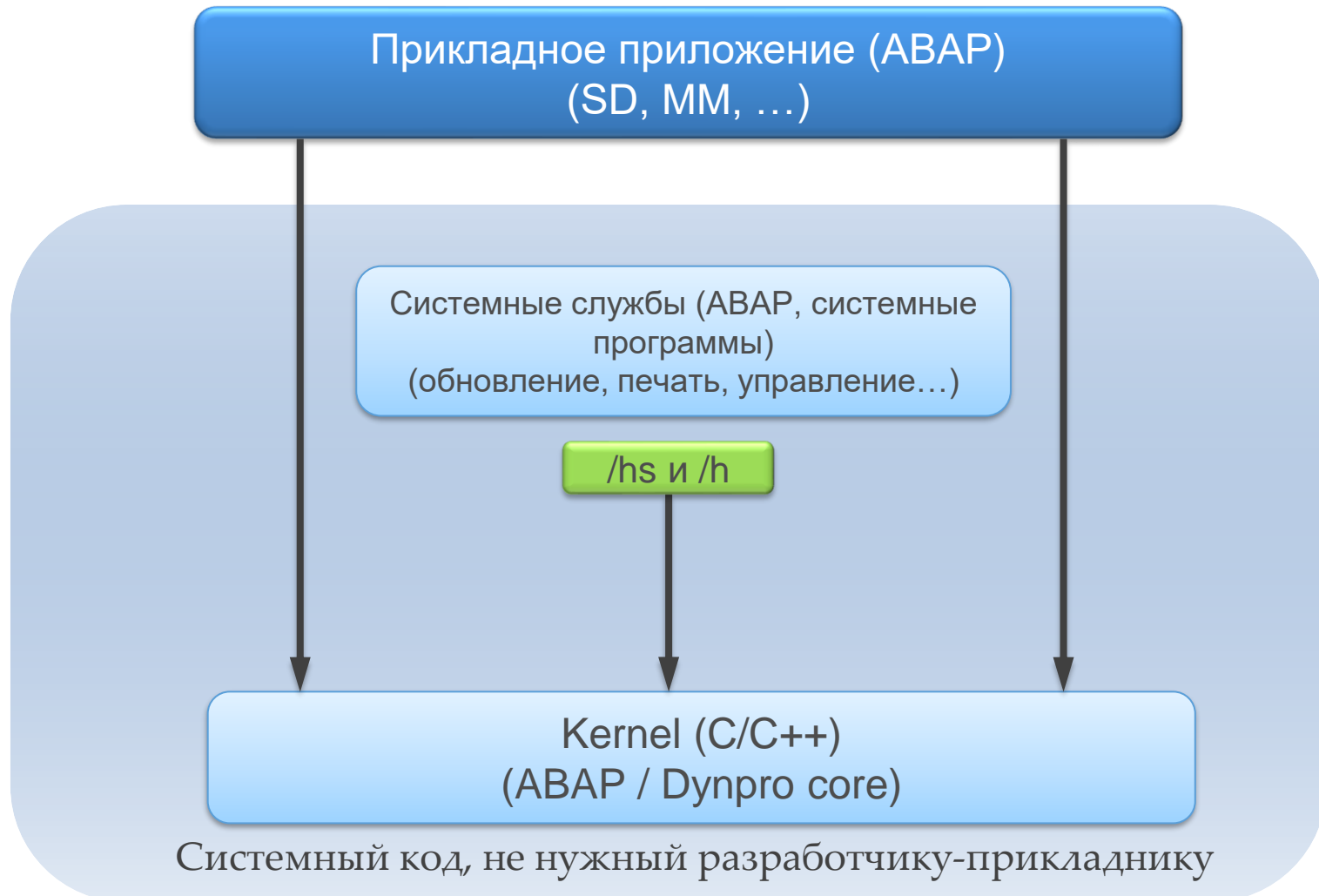
Ускорение поиска и исправления ошибок, имеющих в программе

Отладка с учётом уровня

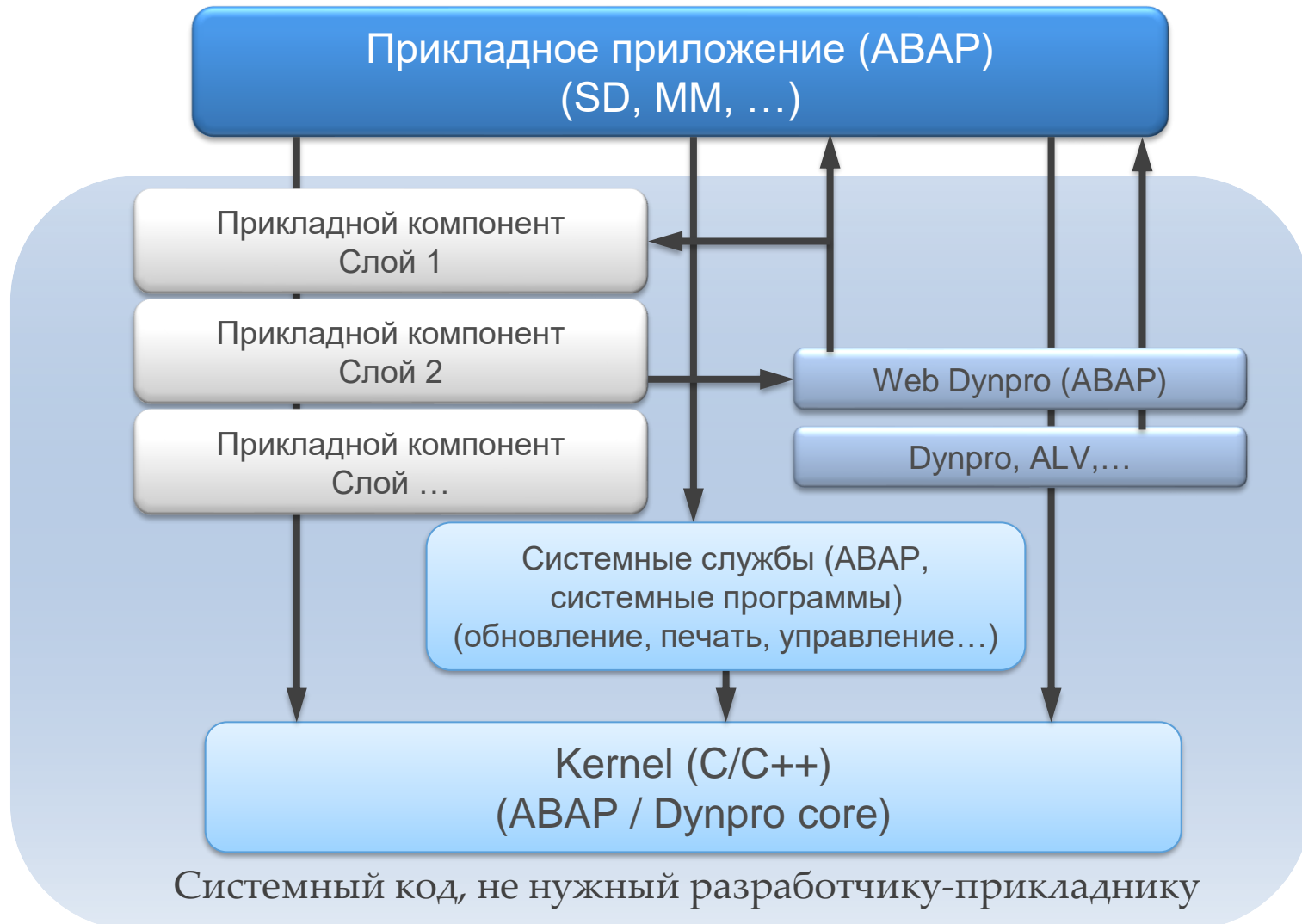
Отладка с учётом уровня. Зачем?



Отладка с учётом уровня. Зачем?



Отладка с учётом уровня. Зачем?



Решение: отладка с учётом уровня

- Определить интересующий код и спрятать мешающий код
- Переходить через уровни или от компонента к компоненту вместо шагов по каждой строчке кода

Графический интерфейс пользователя

Логика приложения

Работа с БД

АВАР-отладчик(1) (Исключая)([REDACTED])

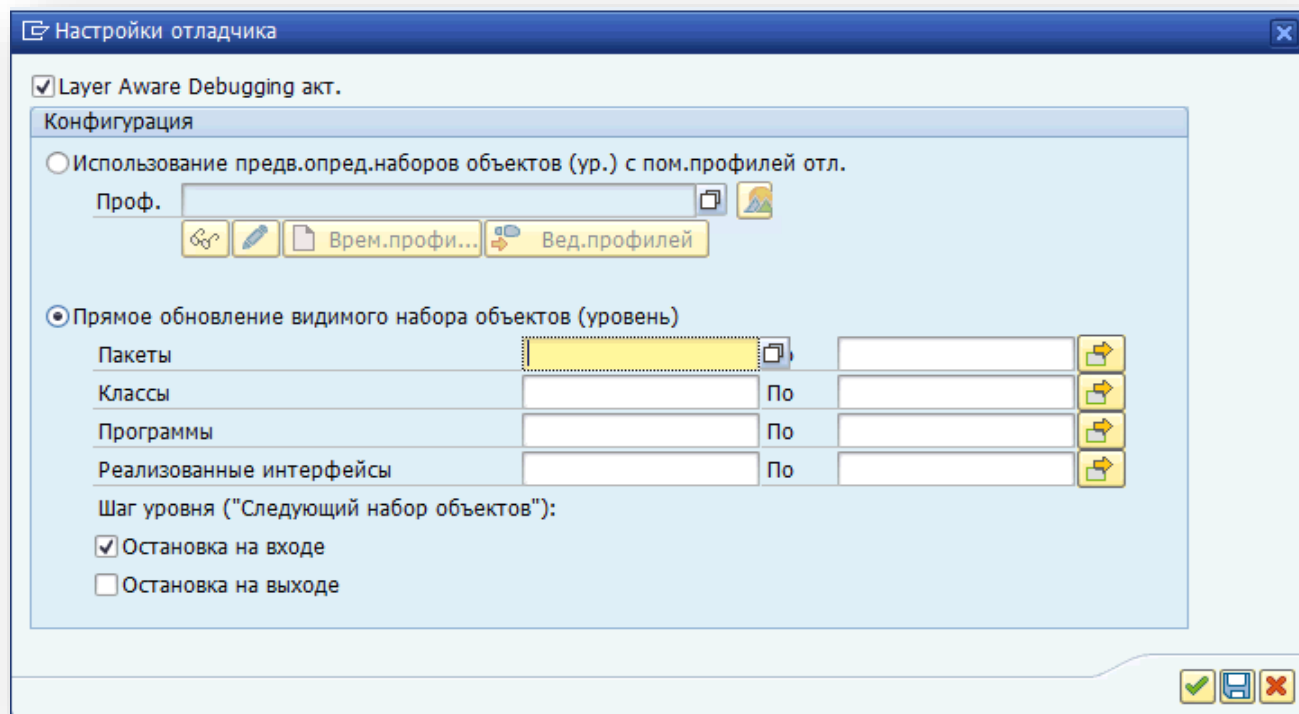
Инкремент | Точка наблюдения | Форма | **Сконфигурировать уровень отладчика**

SAPLSLVC_FULLSCREEN | 0500 / 6 | SY-SUBRC | 0
PAI | | SY-TABIX | 1

Раб. стол | Раб. стол | Раб. стол | Стандарт | Структуры | Таблицы | Объекты | Подр.просмотр | Пров.

```
1  
2 process before output.  
3   module pbo.  
4   *  
5 process after input.  
6 | module pai.
```

Настройка отладчика



Шагаем по наборам объектов

The screenshot shows the SAP ABAP Debugger interface. The title bar reads "ABAP-отладчик(1) - Профиль активен - (Исключая)". The menu bar includes "Следующий набор объектов" (Next set of objects), which is circled in red. Other menu items include "Инкремент", "Точка наблюдения", "Формат", and "Сконфигурировать уровень отладчика". The current object path is "SAPLURL_GENERATION / LURL_GENERATIONU23 / 19" with "SY-SUBRC" set to "0". The function being debugged is "FUNCTION / S_UI_CLASS_DEPENDENCIES" with "SY-TABIX" set to "1". The main window displays a list of variables:

Line	Variable	Type	Value
13	bit	TYPE i,	
14	help	TYPE i,	
15	field_number	TYPE i,	
16	count	TYPE i.	
17			

On the right side, the "Стек AB" (AB Stack) is visible, showing a stack of frames with values 12 and 11.

Шагаем по наборам объектов

```
34
35     endif.
36
37     io_router->attach( iv_template = <ls_entry>-url
38                       iv_handler_class = <ls_entry>-class ).
39 |   endloop.
40 |
41 |   endloop.
42 |
43 | endmethod.
```

ABAP Стр 39 Ст 1

Вход в набор объектов ZDEMO_1

Шагаем по наборам объектов

ABAP-отладчик(1) (Исключая) - HTTP - (...)

Инкремент | Точка наблюдения | Форма | **Активировать Layer Aware Debugging**

SAPLRSEC_CHECKS / LRSEC_CHECKSF05 / 185 SY-SUBRC 0
FORM / GET_EXIT SY-TABIX 0

Раб. стол | Раб. стол | Раб. стол | Стандарт | Структуры | Таблицы | **Объекты** | Подр.просмотр

```
172     CHANGING
173     instance                = c_r_e
174     EXCEPTIONS
175     no_reference             = 1
176     no_interface_reference  = 2
177     no_exit_interface       = 3
178     class_not_implement_interface = 4
179     single_exit_multiply_active = 5
180     cast_error              = 6
181     exit_not_existing       = 7
182     data_incons_in_exit_managem = 8
183     OTHERS                  = 9.
184
185     IF sy-subrc > 1.
186         IF sy-subrc = 8 AND ( cl_rso_comp_medi
187             * That is okay. (Lean system: LOBB not e
```

Стек ABAP и экрана

Ук...	Глу...	В..	Тип события	C
→	43	FORM	FORM	G
	42	FUNCTION	FUNCTION	R
	41	FUNCTION	FUNCTION	RI
	40	METHOD	METHOD	_
	39	METHOD	METHOD	IF
	38	METHOD	METHOD	V
	37	METHOD	METHOD	V
	36	METHOD	METHOD	G

Транзакция SLAD

Пульт управления SLAD: изменение профилей и наборов объектов

Набор объект... ZDEMO_1

Администрир. Критерии

Определение выборок

Имя	Критерий выбора	Вид выборки	Просмотр/изменение выборки
UJ	Пакет с подпакет...	Экран выбора	[иконка]
Z_CODE	Программа/клас...	Экран выбора	[иконка]
UJ_SPEC_CLASSES	Программа/клас...	Экран выбора	[иконка]
UJC	Пакет с подпакетами	Экран выбора	[иконка]
EXIT_HANDLER	Программа/класс	Экран выбора	[иконка]

Соединение объемов выборок (наприм., SEL1 AND SEL2)

```
1) ujc or z_code or exit_handler
```

Выбор программ

Программы Классы Группы функций

Программа [Z+] [иконка]

1

2

3

4

Шаг 1: наборы объектов.

Уровень 1
L1

Логическое выражение
"S1 AND (NOT S2)"

Набор 1
S1

Пакеты P1, P2, P3

Набор 2
S2

Классы CL_1, CL_2

Уровень:

- Объект транспортной системы

Выражение:

- Имена наборов как операнды
- Любые логические операнды

Критерии выбора:

- Пакеты
- Программы / Классы
- ФМЫ
- Реализация интерфейсов

Шаг 2: поведение отладчика

Видимость

Точка
входа

Точка
выхода

<<Остальной код>>



L1 (например, мой пакет)



L2 (например, ALV логика)





35% скидка на любой мастер-класс сессии
по промокоду:

SPACE

- до 28 октября включительно
- только при покупке от физического лица



Никита Калущкий

Моб.: +7 (995) 880-44-45

E-mail: n.kalutskiy@mail.ru

Современная отладка в ABAP

sapland.ru