

Boost Verification Efficiency with VC Execution Manager

Taruna Reddy, Principal Product Manager
Raghuram S Gururaja, Applications Engineering Manager
Synopsys



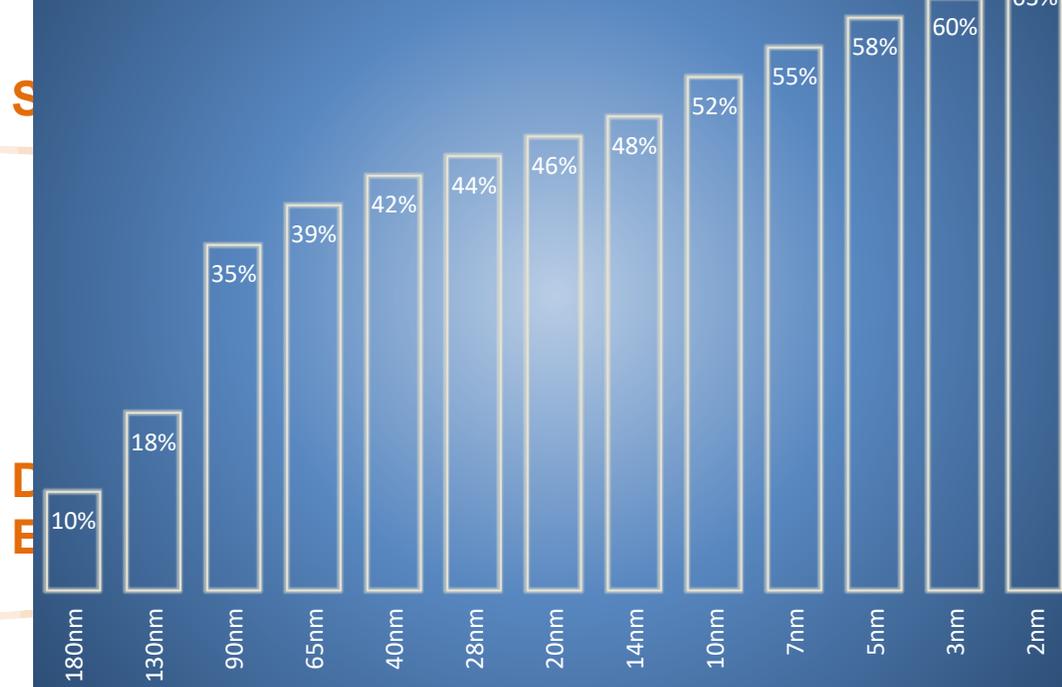
Agenda

- Introduction to VSO.ai (Verification Space Optimization)
- Verdi RDA (Regression Debug Automation)
- VC Execution Manager
- Benefits
- Demo
- Conclusion

The Right First Time Silicon Challenge

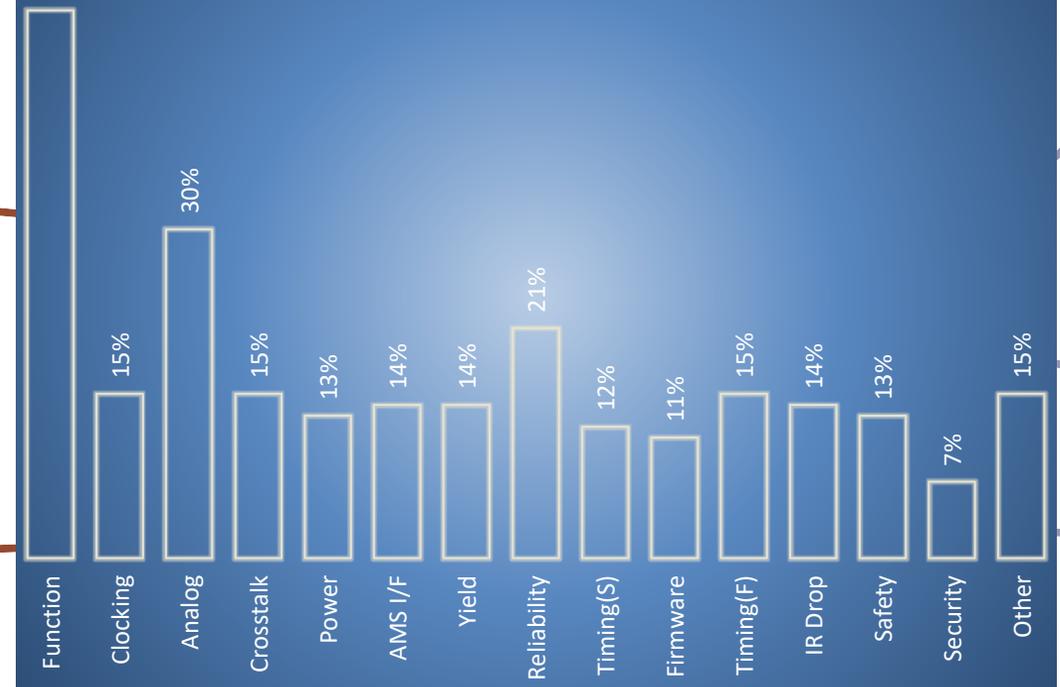
Re-spin Probability

Source: Semico Research Group



Cause of Re-spins

Source: Wilson Research Group 2024

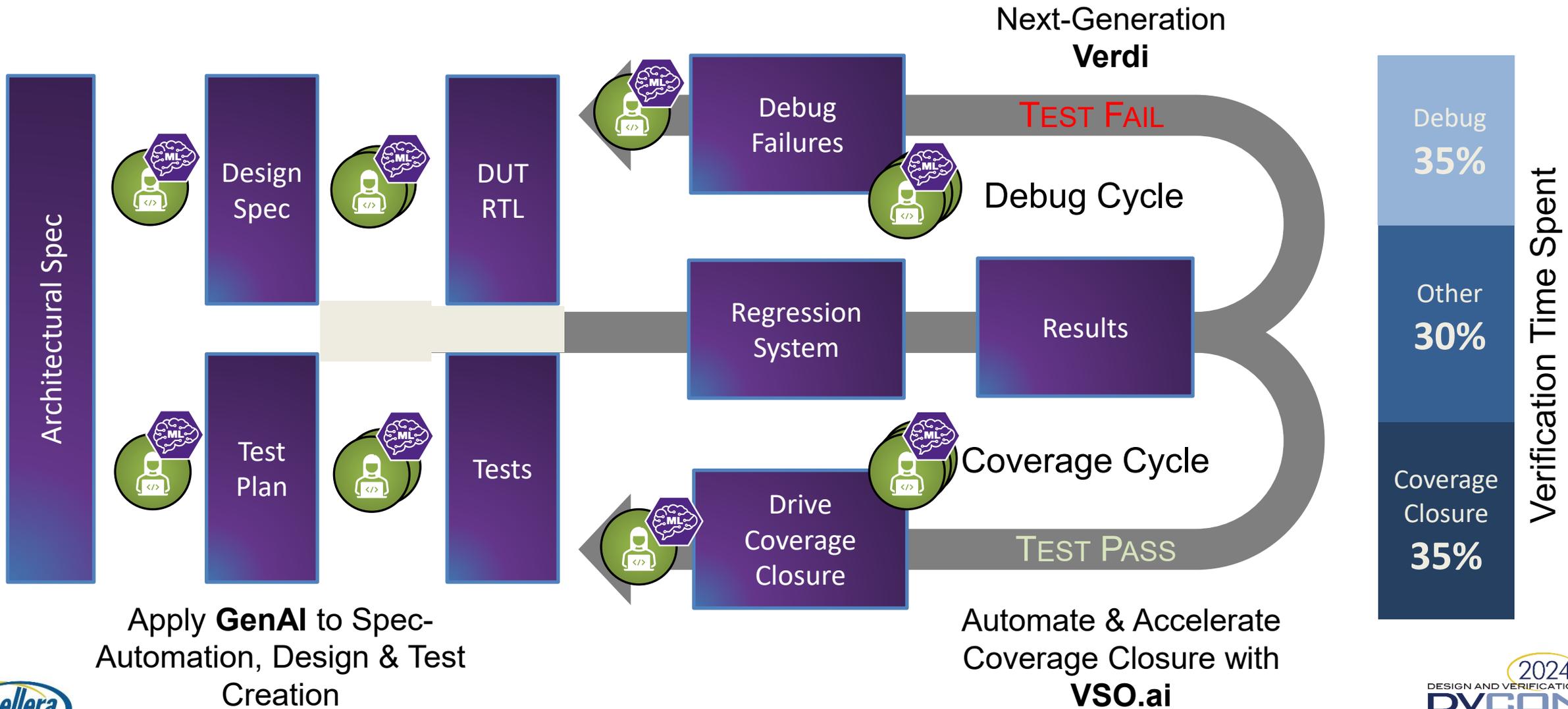


Design size & Verification

Find More Bugs with Limited Time & Resources

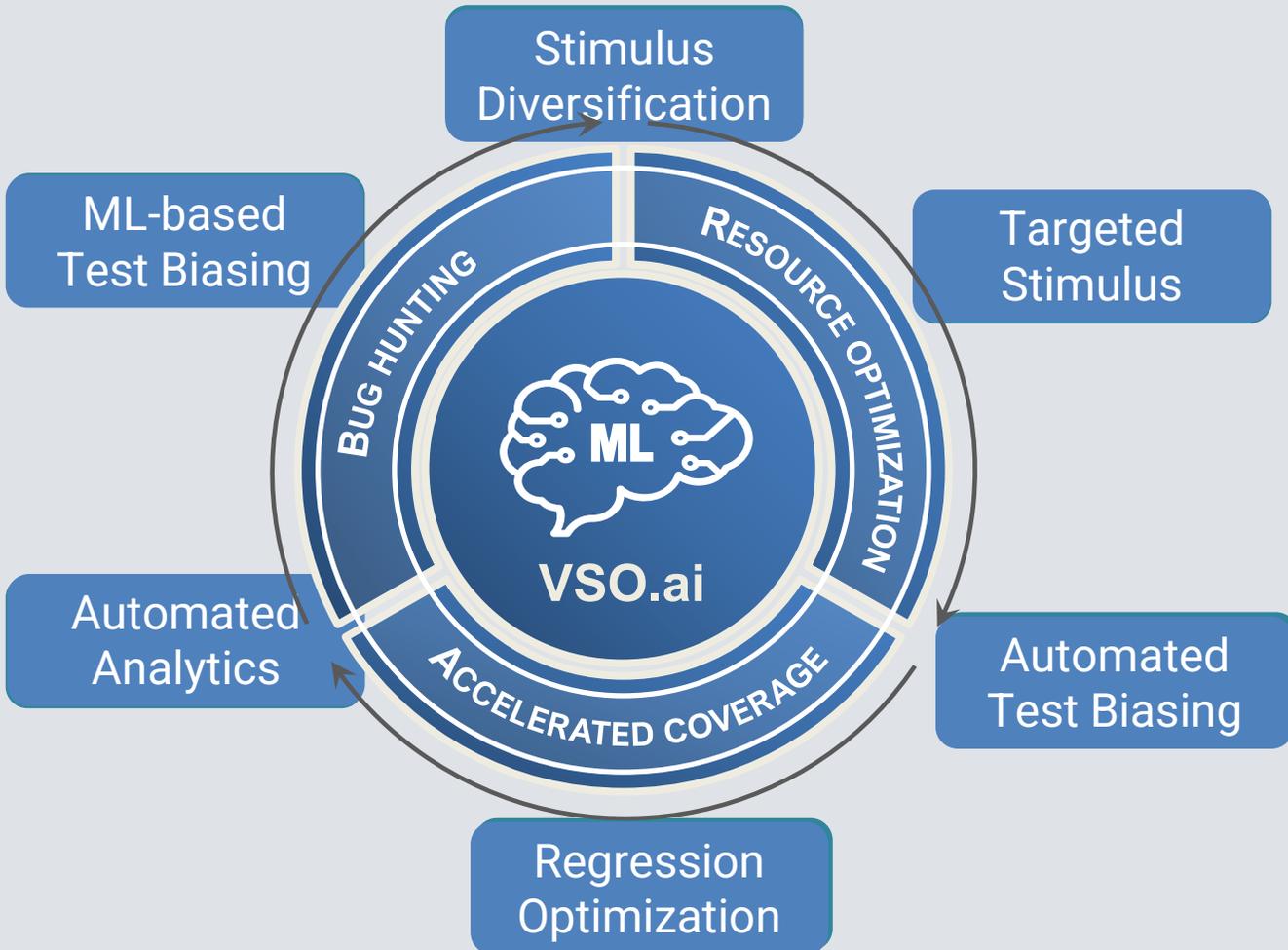
Where Are The Opportunities for Automation?

	Manual Process
	AI-Assisted



AI Driven Verification Space Optimization - VSO.ai

Faster, Higher Coverage Closure & Analytics



Stimulus Diversity: *Uncover bugs earlier while stressing the design*

Productivity Boost: *Advanced dashboards, metrics to analyze to converge on goals*

Improved HW Utilization: *Maximum coverage for the compute budget*

Higher Verification Efficiency: *High ROI tests to achieve coverage targets*

See What Customers Are Experiencing with VSO.ai

REGRESSION

COVERAGE



Automotive
SoC



REDUCED
TAT BY
2x



IMPROVED BY
10%



HIGHER
QoR BY
50%



High-
Performance Data
Center CPU



REDUCED
TAT BY
16x



IMPROVED BY
5%



FASTER TTR IN
1st
ITERATION



High-
Performance
Wireless MCU



REDUCED
TAT BY
10x



IMPROVED BY
20%



2000
RARE COVERAGE
TARGETS IDENTIFIED



Mobile
SoC



REDUCED
TAT BY
8x



IMPROVED BY
3%



2600
ILLEGAL BINS
EXPOSED

VERDI RDA (REGRESSION DEBUG AUTOMATION)

Verdi Debug and Verification Management Platform



Widest-Adopted Debug Solution

Full-featured debug spans gates to software

Infused with AI-based root cause analysis

Integrated with Synopsys/third-party tools

Extendible with user-defined apps

Provides full verification management

Enables IDE extension for VS Code

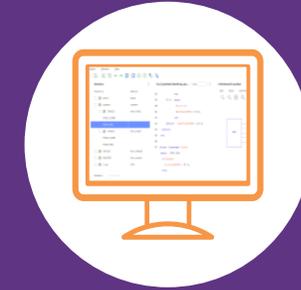
VERDI PLATFORM



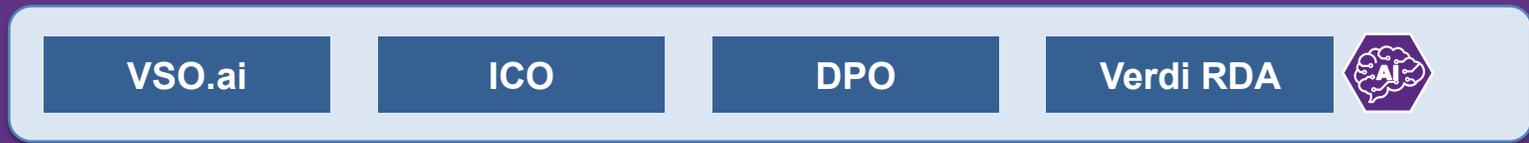
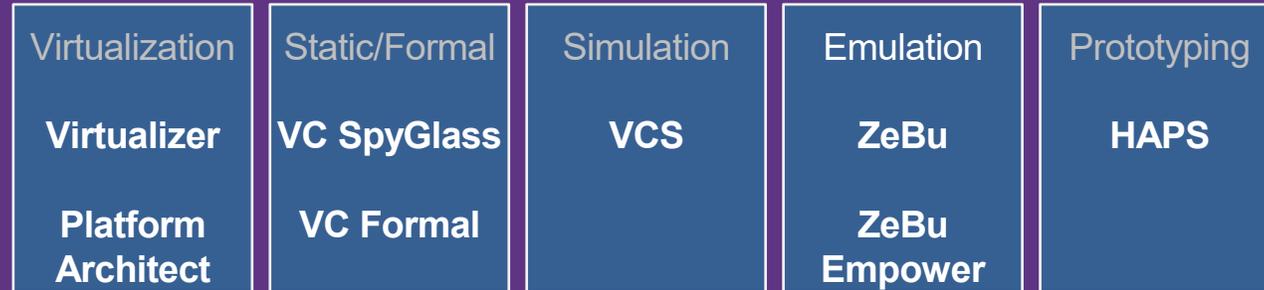
Innovative
Debug



Verification
Management (VMS)

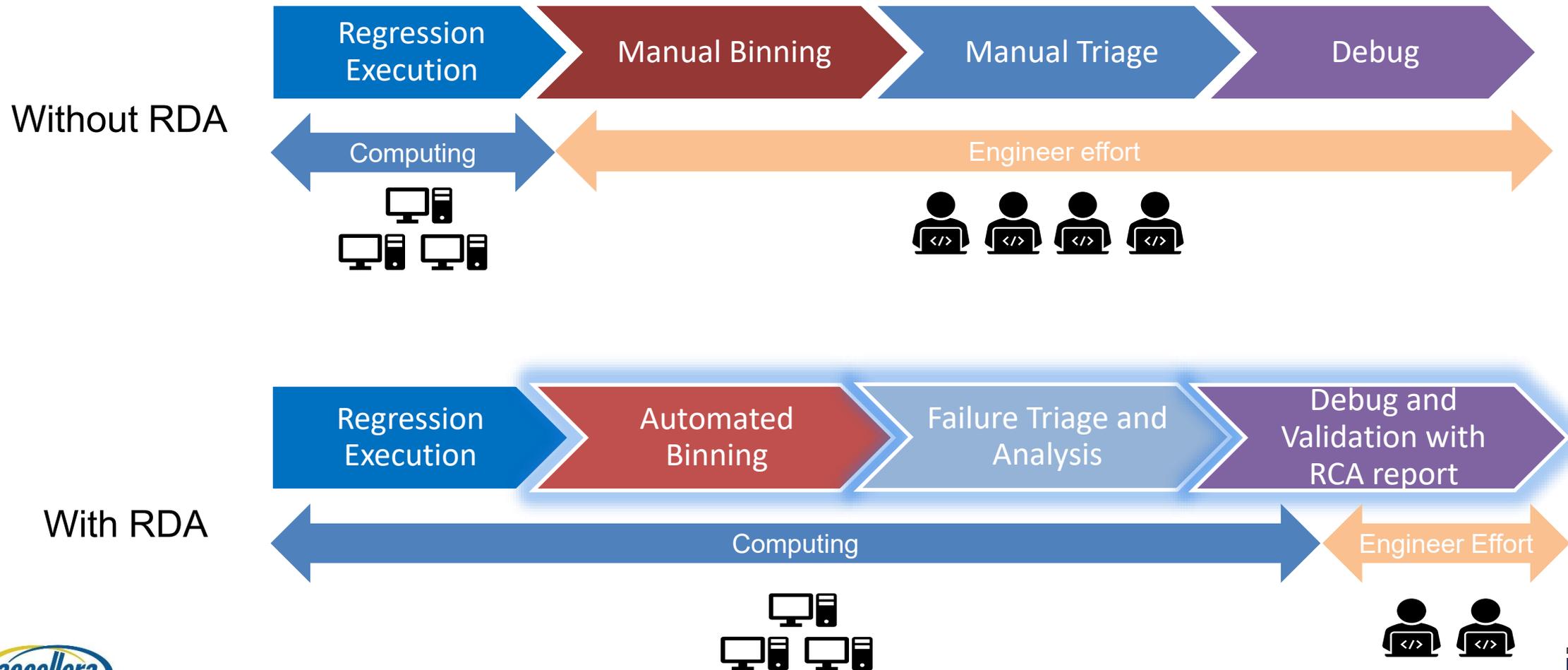


Integrated Development
Environment (IDE)

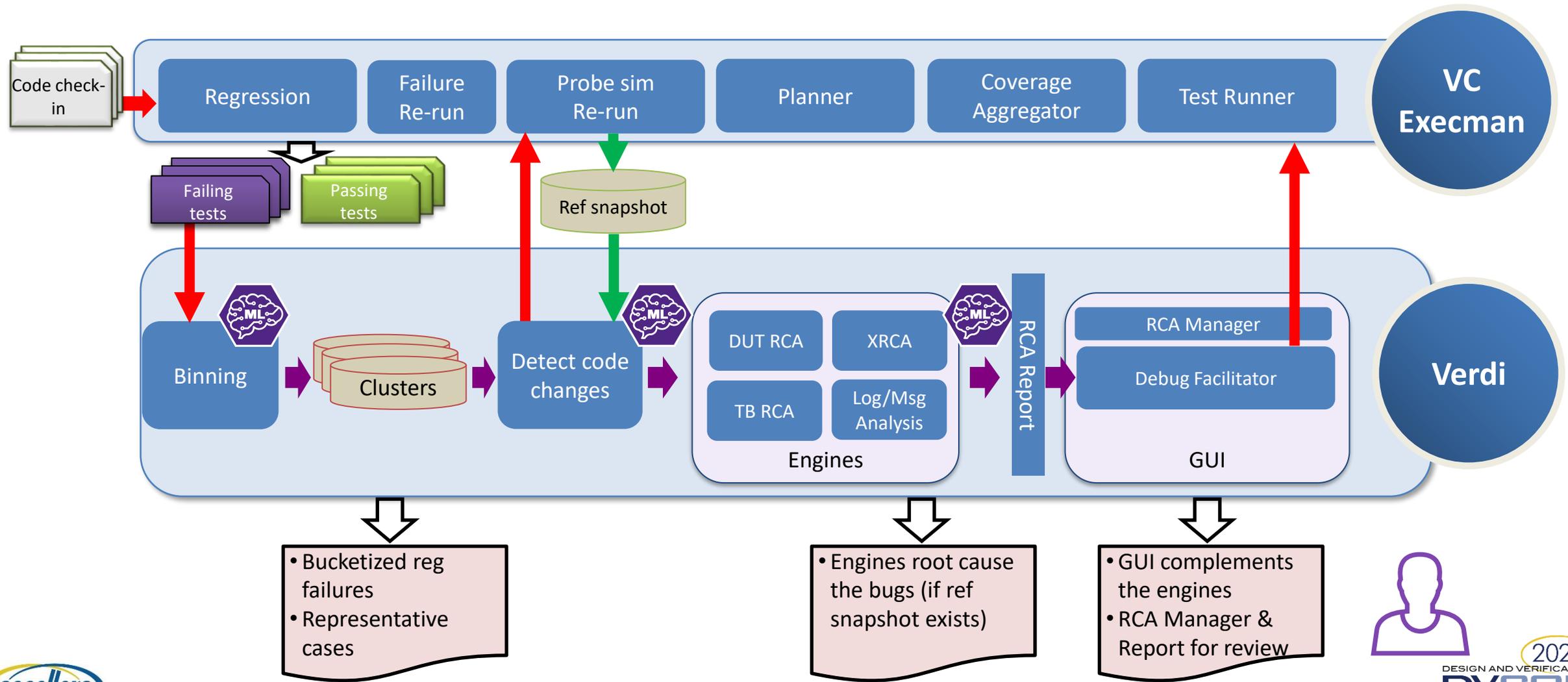


Regression Debug Automation (RDA) Motivation

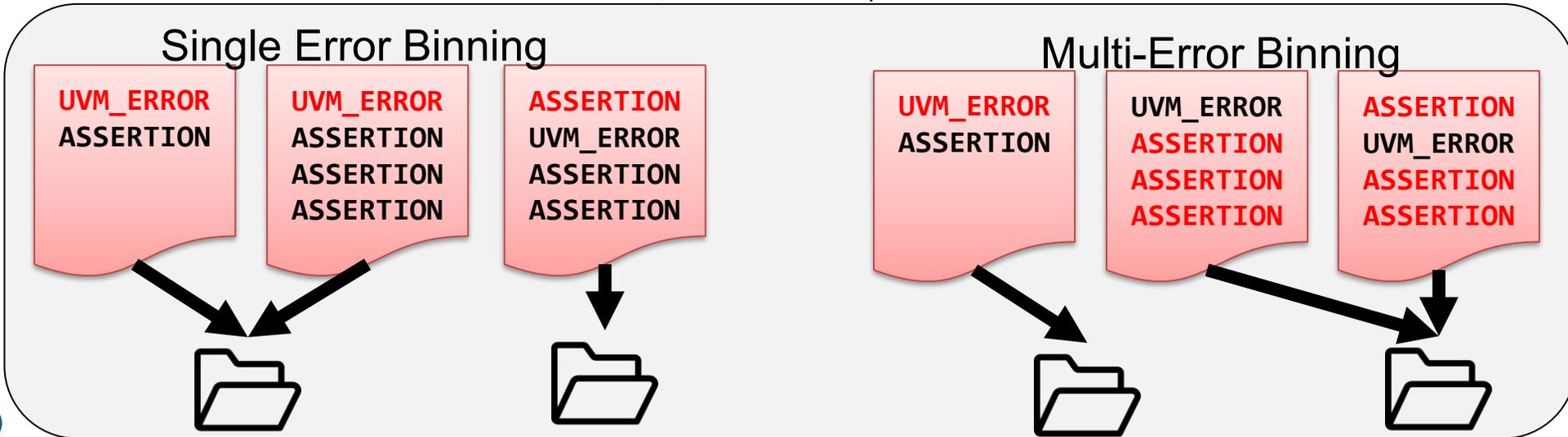
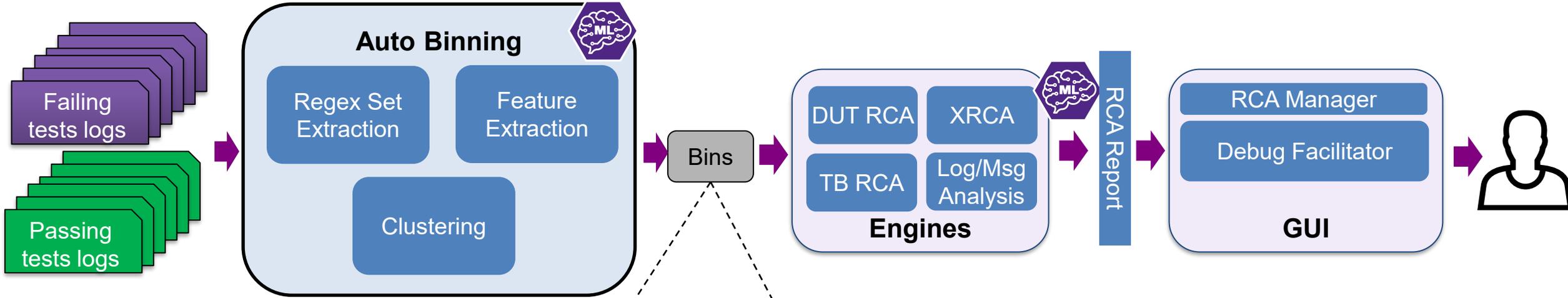
Reduce engineering effort/TAT with AI and advanced RCA technologies



ML-Based, Automated Regression Debug

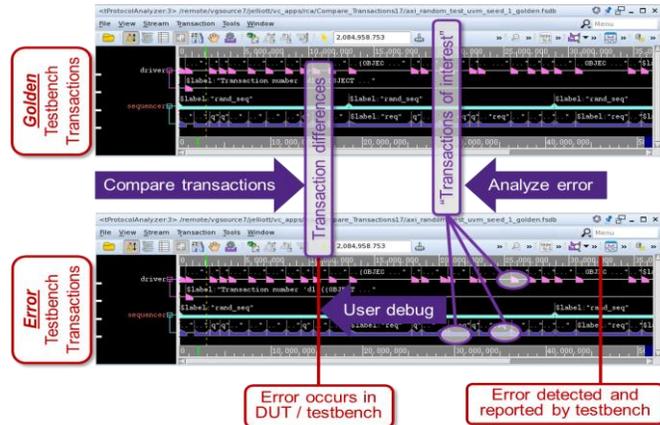


Verdi Regression Binning with ML



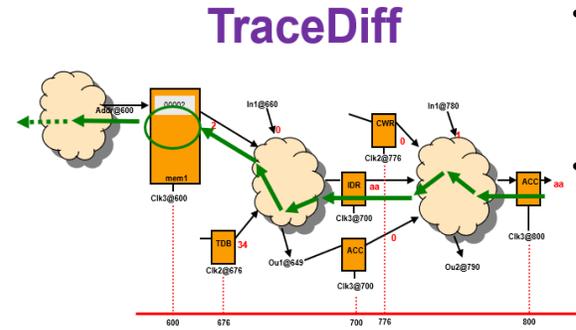
Verdi Root Cause Analysis (RCA) Engines

TBRCA



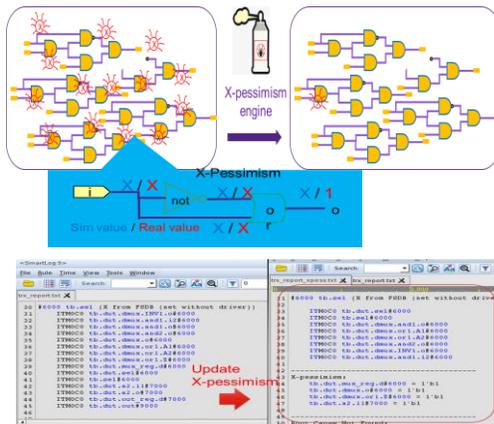
- Transaction Diff – Diff the transaction in the reference vs failing FSDB
- Message Analysis – No ref FSDB required. Analyze info from the “error” message.
- Report transaction of interest linked to the error

DUTRCA



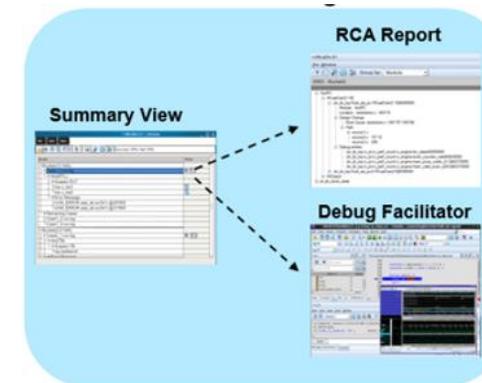
- Adopt roll back mechanism and TraceDiff technology to narrow down DUT problem
- Temporal Flow View to analyze root cause path

XRCA / with X-Pessimism



- Scan X signals in FSDB and trace the root cause of X.
- Handle large amount of X signals in batch mode
- Formal engine to identify X pessimism to remove the noise

Debug Facilitator



- Generates debug data nightly for each bin.
- Facilitates user to debug with “Protocol Analyzer” and “TB Reverse Debug”
- Reduces about 30 ~ 40% debug effort for TB debug

Verdi Next-Gen: Accelerate Debug Automation

Customer Examples



Automotive

**Gate-level
with Many Xs**

60x

XRCA Engine



Video

**DUT
Code Change**

20x

DUTRCA Engine



Server

**Multiple Failing
Assertions**

10x

DUTRCA Engine

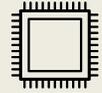


Graphics

**DFT with
X Monitors**

10x

XRCA Engine



Large SoC

**Massive
Log Files**

5x

ML-Binning



Video

DUT Hang

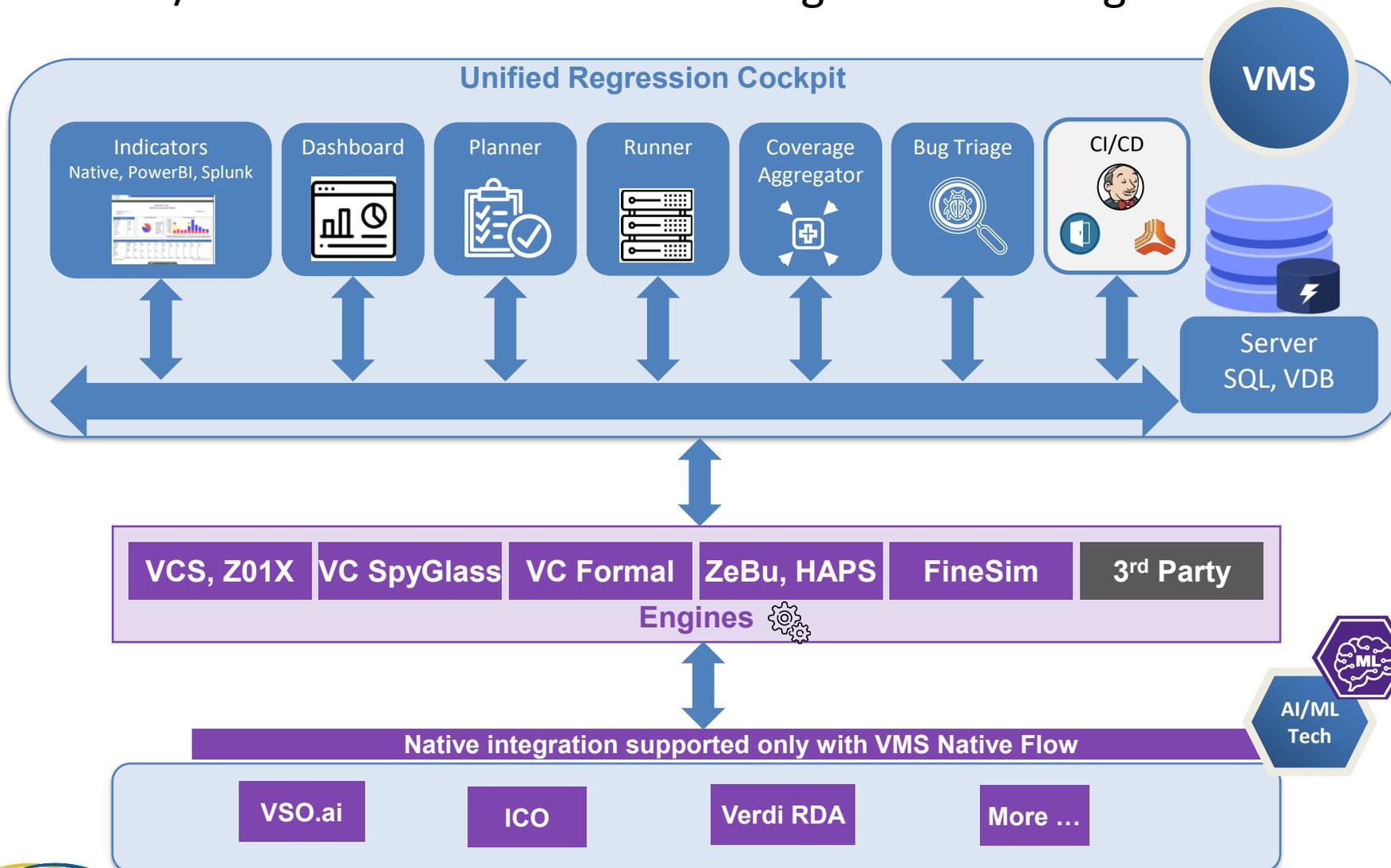
48x

DUTRCA Engine

VC EXECUTION MANAGER

VC Execution Manager

With AI/ML driven Verification and Regression Management



- Cockpit GUI based on Verdi
- Native indicators with open database support for PowerBI & Splunk.
- Comprehensive dashboards
- Scalable multi-user Verification/Coverage Planner
- Unified and extensible Runner
- 24x7 Coverage Aggregation
- Native failure binning, triage and debug assistance
- Unified AI-enabled technologies
- Enables CI/CD methodology support for verification
- Cloud support
- Supports industry standard APIs – Rest, CLI, Python etc.

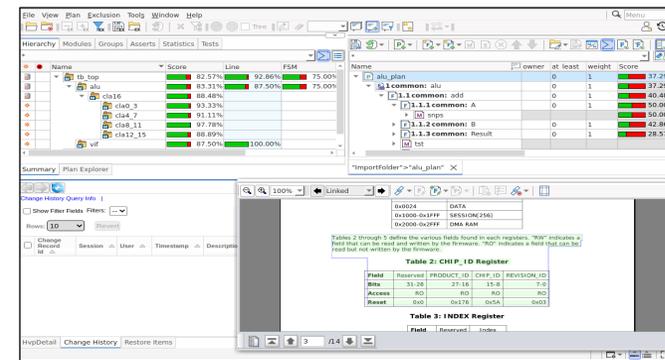
VC Execution Manager – Key Modules

Dashboard



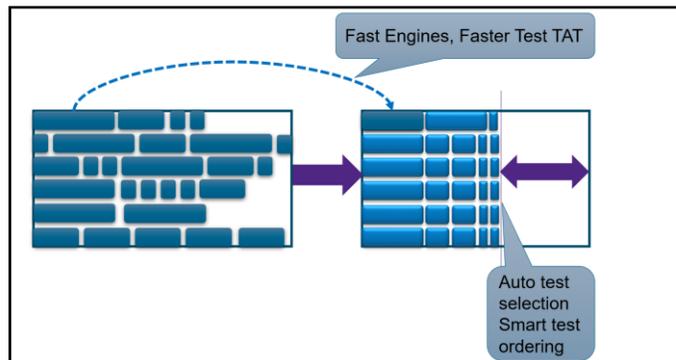
- Test planning, execution & debug, coverage merge and annotation
- Enables verification data-over-time to be mined for analytics

Planner



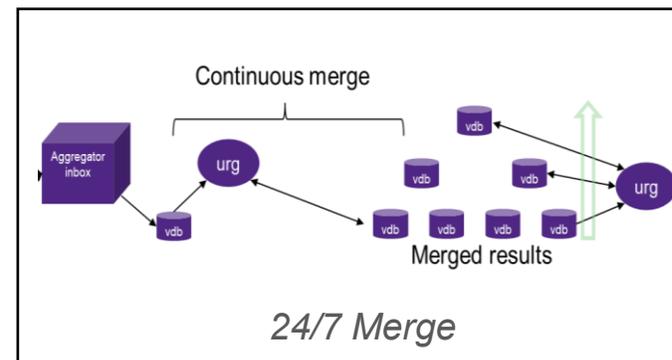
- Multi-user test scheduling/planning
- Supports change history and restore
- API for automated report generation and updates

Runner



- Runs and optimize regressions
- Order tests to eliminate long tail

Coverage Aggregator



- Continuously merges incoming coverage
- Integrated tagged VDB from ad hoc regression runs
- Can generate moving window merge VDB

Dashboard Users View – Usability

The top screenshot shows a test session summary with columns for Name, Status, Start Time, RTime(s), Hostname, Log File, Comment, Cmltime, First Error Time, Tags, Action, and Test Trend. It lists several test sessions with their respective statuses and times.

The middle screenshot displays a hierarchical tree view of test modules and their coverage scores. For example, 'alu_pkp_alu_monitor_add_trans' has a score of 33.33%, while 'operand_B' has a score of 35.71%.

The bottom screenshot provides a detailed view of a specific test item, showing its code structure and coverage analysis. It includes a 'Cover Group Item' table with columns for Name, Score, Instances, UAC, U, C, and X.

Engineer View



The top screenshot is a line chart titled 'Execution Manager' showing test trends over time. The Y-axis represents 'PERCENTAGE' from 60% to 100%. The X-axis represents 'INDEX'. Four data series are shown: /Wishbone:Assert (green), /Wishbone:Cond (blue), /Wishbone:Group (orange), and /Wishbone:Line (yellow).

The bottom screenshot is a table view of test sessions. The columns are: Name, #Sessions, Latest Session, Tot, P, F, W, A, U, Summary, Created, Test Trend, and Plan Trend.

Name	#Sessions	Latest Session	Tot	P	F	W	A	U	Summary	Created	Test Trend	Plan Trend
modes	42	session_thielges_20180613_135341	7	3	4	0	0	0	43% 57%	2018/01/17 23:37:50	🇮🇹	🔄
etrend	161	session_thielges_20180613_111627	10	10	0	0	0	0	100%	2018/02/13 14:40:39	🇮🇹	🔄
alu_uvm	137	session_thielges_20180612_145759	7	3	4	0	0	0	43% 57%	2014/12/22 16:22:01	🇮🇹	🔄
abc	1	session_thielges_20180501_150822	0	0	0	0	0	0		2018/05/01 15:08:22	🇮🇹	🔄
saves_vdb	13	session_thielges_20180424_231913	10	3	7	0	0	0	30% 70%	2018/04/24 17:11:00	🇮🇹	🔄
merge_ex	3	session_thielges_20180424_170132	10	3	7	0	0	0	30% 70%	2018/04/24 16:57:21	🇮🇹	🔄
srivatsa	2	session_thielges_20180420_110122	1	1	0	0	0	0	100%	2018/04/20 10:54:10	🇮🇹	🔄
filespew	2	session_thielges_20180306_153626	1	0	1	0	0	0	100%	2018/03/06 15:32:52	🇮🇹	🔄
two_builds	1	session_thielges_20180221_121418	7	1	6	0	0	0	14 86%	2018/02/21 12:14:19	🇮🇹	🔄
simple	4	session_thielges_20180216_122156	9	3	6	0	0	0	33% 67%	2018/02/16 09:50:39	🇮🇹	🔄

Management View

- Verification plans created/viewed by multiple users

- Organizes content by projects and projects hierarchically

- Scalable for large designs and users and cloud compatible

- Smart editor with rich text support, inline images, tables; bulk edits; API

Planner

DB based Verdi Hierarchical Verification Planner (HVP)

The screenshot displays the Synopsys Planner application interface. The top window shows a hierarchical tree view of verification plans. The left pane shows a tree structure starting with 'tb_top', which contains 'alu', which in turn contains several 'cla' (class) folders like 'cla16', 'cla0_3', 'cla4_7', 'cla8_11', and 'cla12_15', and a 'vif' folder. The right pane shows a detailed view of the 'alu_plan' folder, listing its contents: '1 common: alu', '1.1 common: add', '1.1.1 common: A', '1.1.2 common: B', '1.1.3 common: Result', and 'tst'. Each entry includes a score and a progress indicator.

The bottom window shows a 'Change History Query Info' panel with a table of change records. Below this, there is a detailed view of a register definition, including a table for 'Table 2: CHIP_ID Register' and 'Table 3: INDEX Register'.

Table 2: CHIP_ID Register

Field	Reserved	PRODUCT_ID	CHIP_ID	REVISION_ID
Bits	31-28	27-16	15-8	7-0
Access	RO	RO	RO	RO
Reset	0x0	0x176	0x5A	0x03

Table 3: INDEX Register

Field	Reserved	Index
0x0024	DATA	
0x1000-0x1FFF	SESSION[256]	
0x2000-0x2FFF	DMA RAM	

Planner - Details

- **Database** backed application enables real time collaboration
- **Typed features** ensures that all plans satisfy project template requirements
- **Rich text fields** to describe detailed descriptions
- **Project-wide queries and bulk updates**
- **Change history and restore**
- **API** for automated report generation and updates
- **Verdi UI** (Linux) and **Web UI** (all platforms)

The screenshot displays the Synopsys Planner application interface with several key components highlighted by blue callout boxes:

- Plan Explorer:** A tree view on the left showing the project structure, including folders like 'ImportFolder', 'Memsys Verification Plan_incl', and 'jukebox_top_plan_incl'.
- Plan Detail:** A central pane showing a list of features under the 'jukebox_coin_handler' plan, such as '1 FT_1: coin_handler_1', '2 FT2: coin_handler_2', '3 jukebox_CD_handler', '4 FT2: Feature_1', '5 jukebox_station', and '6 FT2: table1-1'.
- Change History:** A table at the bottom showing feature identifiers, attribute limits, and child limits.
- Project Search:** A search bar at the bottom left.
- Restore Items:** A button at the bottom center.
- Richtext Viewer:** A window on the right showing a rich text editor with a toolbar and a preview area containing a diagram.

Identifier	Attribute Limit	Child Limit
FT2	<All>	FT_1,common,me...
FT_1	<All>	<All>
common	<All>	common,measure...

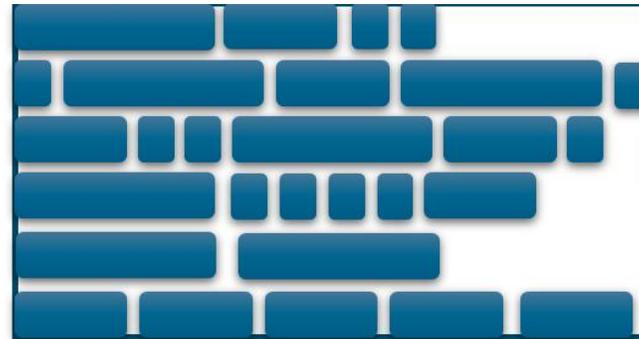
Synopsys Confidential Information

Optimize Regressions

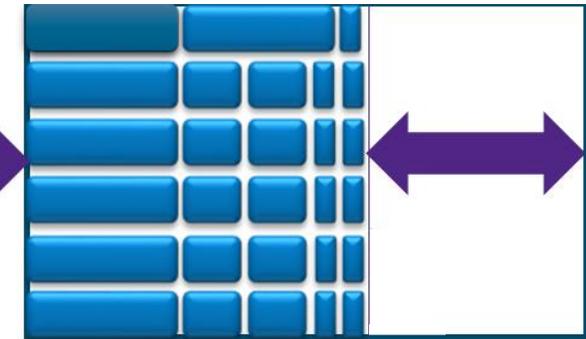
Runner

- Reduce regression TAT and resources
- Schedules tests based on history to eliminate long tail
- Improves compute resources utilization, reducing costs
- Native cloud support

Scheduled Tests (Before)



Scheduled Tests (Optimized)

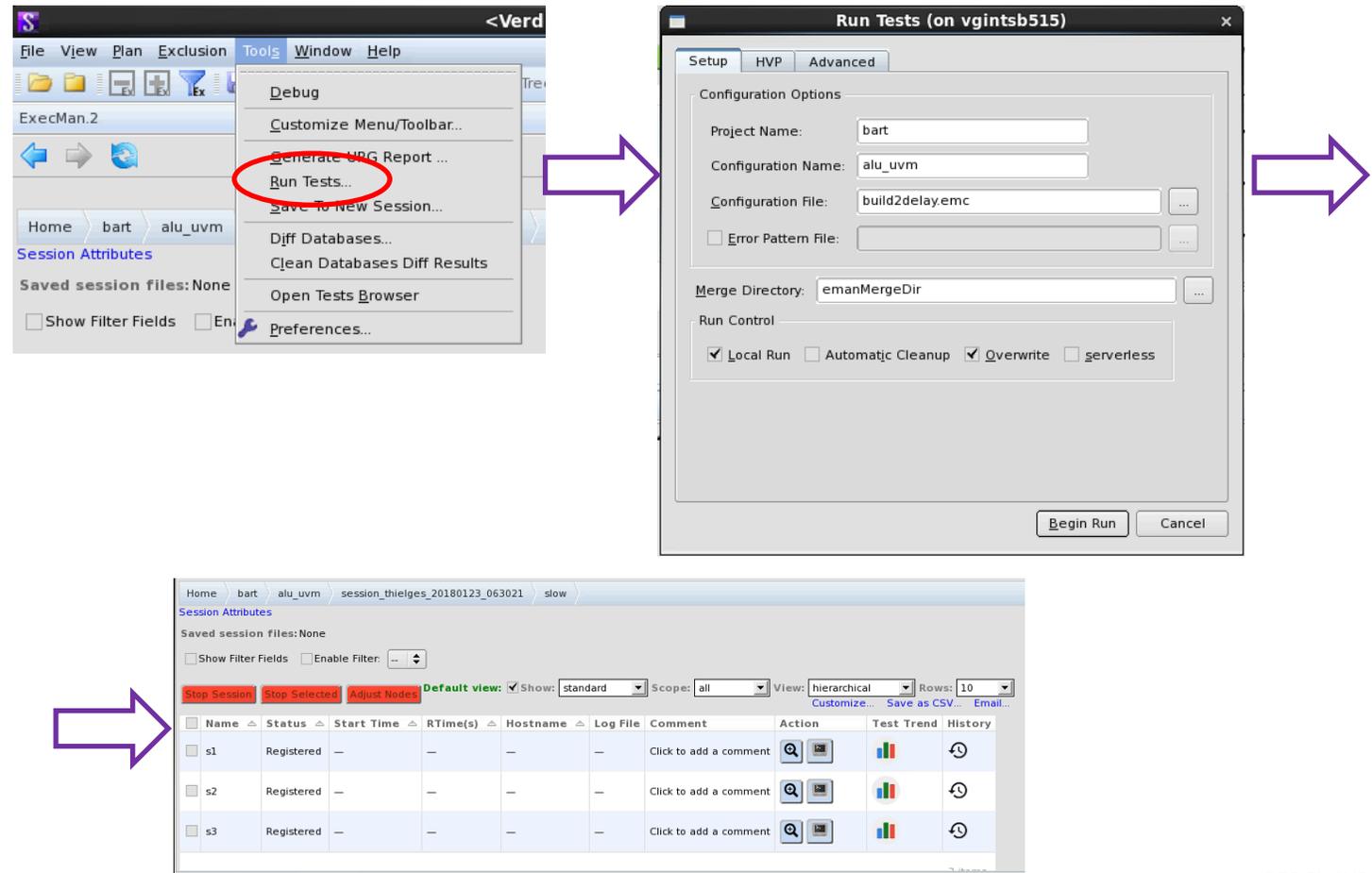


Considers: coverage, engine speed, smart test ordering

Runner - Launching Regressions

Methods

- Interactive
 - Easiest way to launch regressions
 - Includes live controls to kill, rerun, and adjust sessions
- Batch
 - Useful as part of complex scripts or launched as cron jobs



Coverage - Beyond Simulation

VC Formal example

- Coverage from additional verification tools can be incorporated

- Annotate VC Formal results in the coverage database

- Display VC Formal assertion status in the coverage report
 - Verdi coverage and URG

- Measure VC Formal assertion status in HVP

Assert	Type	Success/M	Attempt	Failure	Incomplete	Vacuous	Category	AllMatch	Severity	FirstMatch	FVtype	FVstatus	FVdepth
fsm.a1	Assertion	1	0	0	0	---	0	---	0	---	assert	proven	
fsm.a2	Assertion	0	0	0	0	---	0	---	0	---	assert	falsified	65
fsm.a_complete_frame	Assertion	1	0	0	0	---	0	---	0	---	assert	proven	
fsm.a_loop_break	Assertion	0	0	0	0	---	0	---	0	---	assert	inconclusive	70
fsm.a_onehot	Assertion	1	0	0	0	---	0	---	0	---	assert	proven	
fsm.a_trans	Assertion	0	0	0	0	---	0	---	0	---	assume	proven	
fsm.c1	... Property	1	0	---	0	0	0	---	0	---	cover	proven	
fsm.c2	... Property	0	0	---	0	0	0	---	0	---	cover	inconclusive	70
fsm.c_blk_cnt	... Property	1	0	---	0	0	0	---	0	---	cover	covered	1
fsm.c_onehot	... Property	1	0	---	0	0	0	---	0	---	cover	covered	1
fsm.c_trans	... Property	0	0	---	0	0	0	---	0	---	cover	covered	1

Merge Name	Status	Spool Log(s)	URG Report	Score
merged.vdb	merge passed	60.log	urgReport	51.59

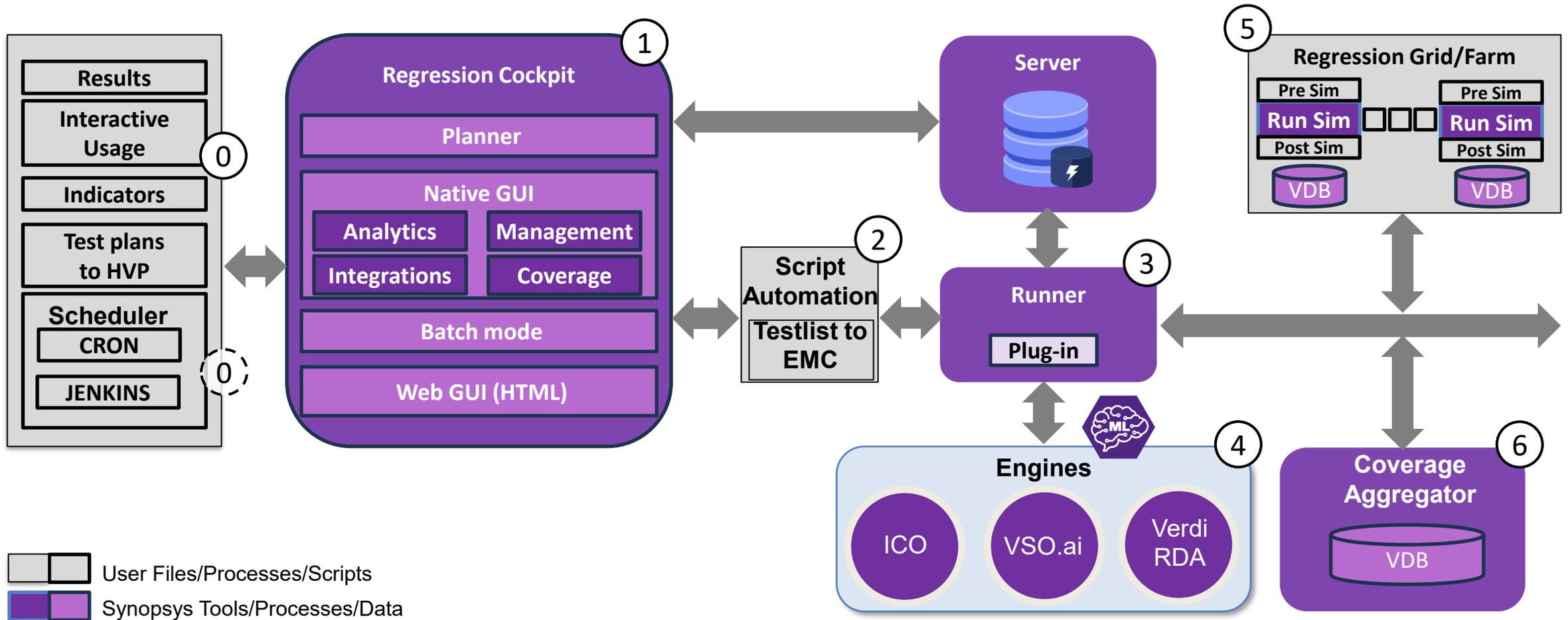
Scores for Verification Plan

SCORE	ASSERT	FVAssert	NAME
36.84	31.58	6/19	42.11 top

NAME	SCORE	ASSERT	FVAssert	fvassert_expected_status	fvassert_expected_mindepth	fvassume_status
top	36.84	31.58	6/19	42.11	proven	-1 notreviewed
f_default	37.50	37.50	6/16	37.50	proven	-1 notreviewed
f_expected_assigned	25.00	0.00	0/2	50.00	inconclusive	50 notreviewed
f_assume	50.00	0.00	0/1	100.00	proven	-1 reviewed

AI/ML INTEGRATION AND BENEFITS

Integrated View: VC ExecMan with VSO.ai & Verdi RDA



Unified Verdi VMS with AI/ML Technologies

Optimized verification with reduced resources, risks and maintenance

Ease of Use and Data Analytics via Integration

- Faster rollout to project teams (save up to 12 months)
- Seamless integration with native solutions like VSO.ai
- Unified cockpit solution for regression results, data analytics, optimization outcomes and health of ML model

Reduced Risk, Simpler Maintenance

- Reduces incremental support overhead from weeks to days
- One DB/server (vs. multiple machines/DB/servers for hybrid systems with glue logic)
- Up to 2X reduced hardware costs

Regular AI/ML Rollouts and Updates

- 2-10X higher QoR with integrated ML and verification technologies when having full data access/control
- Immediate deployments of updates of VSO.ai, RDA, DPO, ICO and VC Formal rollouts (vs. months of integration effort)

DEMO

Conclusion

One-stop solution to ease the complexity of diverse verification tasks

Accelerates the deployment of advanced AI/ML verification and debug technologies

Scales to accommodate multiple users and manage regressions seamlessly

Results made simple with a user-friendly, graphical view of results