Conference Insights Engine:

Slate Summit 2026



The Innovation Forge

David Dysart

Contents

Preface 3

Reading Paths 5

Chapter 1 - Decision Infrastructure 8

Chapter 2 - The Query Ladder 17

Chapter 3 - From Signals to Action 28

Chapter 4 - Governed Personalization 41

Chapter 5 - Events and Portals as Signal Systems 53

Chapter 6 - The Hidden Operating System 64

Chapter 7 - The Build Is Not Done Until It Can Be Taught 76

Chapter 8 - The Next Slate Practitioner Stack 88

Chapter 9 - The Work After The Summit 102

Preface

The Conference Insights Engine began with a simple question: what does Slate Summit reveal when the sessions are read together?

That's what I tried to answer for Slate Summit 2026. It honors the work presenters shared, traces patterns across sessions, and turns those patterns into practical frameworks for admissions, enrollment, and Slate teams.

The chapters are interpretive. They connect ideas that appeared in different rooms, different decks, and different institutional contexts. Read straight through, the collection follows an arc: from better decisions, to better queries, to better signals, to better governance, to better teaching, to the practitioner skills now required by the work.

The goal is usefulness. A chapter has done its job if it helps a team ask a sharper question, clean up a process, teach a teammate, improve a campaign, retire stale logic, or see a Slate instance more clearly.

This is also a guide for action. Readers can move through it in order, or enter through the problem in front of them. A reporting problem may point first to decision infrastructure and the query ladder. A communication problem may point first to signal design and governed personalization. Event and portal work may point first to the signal system frame. An instance that feels hard to maintain may point first to the hidden operating system. Staff training may point first to the chapters on teachable builds and the practitioner stack.

The Summit is full of local examples. This collection looks for reusable meaning inside them. It asks what the community is learning together: how to build systems that are clearer, more governed, more teachable, and more connected to the decisions offices actually make.

Use the collection as a set of chapters, a field guide, or a planning tool.

Read it in order if you want the full argument. The sequence starts with the decision a build supports, then moves into the skills and structures that make the build trustworthy. It ends with a practical adoption model for turning conference learning into office practice.

Use it by problem if your team has a specific need. Each chapter gives a frame, a set of questions, and a practical artifact that can be adapted to local work. The point is translation. A Summit session may show what one office built. A written chapter should help another office decide what version of that idea fits its own context.

Use it for staff development if your team is trying to build shared Slate capacity. Several chapters can become training topics, workshop prompts, documentation exercises, or project review standards.

Use it for leadership conversations if your office needs better language for the work behind the work. Many of the most valuable Slate projects are difficult to explain because they sit between technology, operations, strategy, data, and staff learning. The chapter titles give teams handles for that work.

Each chapter includes source notes. Those notes point readers back to the Summit sessions that shaped the synthesis. The community around Slate makes it great, so let’s forge something extraordinary together.

Reading Paths

Chapter 1: Decision Infrastructure
This opening chapter argues that Slate work should be judged by the decision it improves. A dashboard, model, campaign, portal, or query has value when it changes what people understand, what they do next, and how they review the result. The chapter introduces a five question test for connecting evidence, ownership, action, and review.

Chapter 2: The Query Ladder
This chapter turns query literacy into a shared learning path. It moves from the question being asked, to row grain, bases, direct paths, many related rows, advanced comparisons, and reusable assets. The core argument is direct: if teams cannot explain how a population is built, they cannot trust the work that depends on it.

Chapter 3: From Signals To Action
This chapter examines the difference between collecting engagement data and using signals responsibly. It introduces the Signal To Action Matrix, which asks what a behavior suggests, what it fails to prove, who owns the response, what action is appropriate, what should be suppressed, and how the team will measure whether the action helped.

Chapter 4: Governed Personalization
This chapter reframes personalization as a governed communication system. Strong personalization depends on clear audiences, source definitions, content blocks, templates, Liquid, channels, suppressions, testing, reporting, ownership, and review. The main artifact is the Communications Governance Map.

Chapter 5: Events And Portals As Signal Systems
This chapter treats events and portals as designed experiences that create evidence. Registration, attendance, schedule choice, portal activity, family behavior, follow through, and missing action all carry meaning when they are interpreted with care. The chapter introduces the Event Signal Loop.

Chapter 6: The Hidden Operating System
This chapter looks beneath visible Slate screens to the rules, workflows, assignments, permissions, naming conventions, exceptions, patches, and quiet automations that shape daily work. It introduces a process trace and an operating brief for making hidden logic easier to explain, review, and retire when needed.

Chapter 7: The Build Is Not Done Until It Can Be Taught
This chapter argues that a Slate build is complete when the right people can operate it, explain it, maintain it, and review it without guessing. It introduces the Slate Teaching Brief, role based training levels, teach back, documentation ownership, and staff practice as part of implementation.

Chapter 8: The Next Slate Practitioner Stack
This chapter describes the broader capability profile emerging from advanced Slate work. The next practitioner needs query skill, AI evaluation, data architecture, governance, reporting sense, event

and portal design, documentation habits, training judgment, and cross unit translation. It includes a Practitioner Stack Self Assessment and a one quarter development plan.

Chapter 9: The Work After The Summit
This capstone chapter turns the full series into an adoption model. A team can start with one decision, one process, one signal, one campaign, one hidden object, one teaching artifact, and one practitioner skill. The chapter gives teams a thirty day rhythm for turning Summit learning into office practice.

For **Leaders**

Start with Decision Infrastructure. Then read Governed Personalization, The Build Is Not Done Until It Can Be Taught, and The Work After The Summit.
This path helps leaders connect Slate projects to decisions, owners, review habits, staff capacity, and conference follow through.

For Slate Captains And Senior Builders
Start with The Query Ladder. Then read From Signals To Action, The Hidden Operating System, and The Next Slate Practitioner Stack.
This path focuses on the structures that make advanced builds reliable: population logic, signal definition, hidden rules, governance, maintenance, and practitioner growth.

For Communications Teams
Start with Governed Personalization. Then read From Signals To Action, Events And Portals As Signal Systems, and Decision Infrastructure.
This path helps teams move from clever messaging to governed communication systems with clear audience logic, signal use, suppression rules, testing, and review.

For Events And Visit Teams
Start with Events And Portals As Signal Systems. Then read From Signals To Action, Governed Personalization, and The Work After The Summit.
This path treats events as more than attendance counts. It helps teams define what event behavior means, what follow up should happen, and how event learning should feed future decisions.

For Reporting And Analytics Teams
Start with The Query Ladder. Then read Decision Infrastructure, From Signals To Action, and The Next Slate Practitioner Stack.
This path emphasizes row grain, evidence quality, decision use, signal interpretation, and the judgment needed around models, dashboards, and AI assisted analysis.

For Staff Training And Enablement
Start with The Build Is Not Done Until It Can Be Taught. Then read The Hidden Operating System, The Next Slate Practitioner Stack, and The Work After The Summit.
This path helps teams turn local expertise into shared practice through training briefs, documentation ownership, teach back, and staff development plans.

Chapter 1 - Decision Infrastructure

There is a kind of Slate build that feels impressive when it launches and confusing six months later.

A dashboard has twenty five numbers on it. Everyone agrees it looks useful. A few people open it every week. Someone mentions it in a meeting. Then someone asks the question that matters: what decision does this dashboard actually change?

The room gets quiet.

Or there is a model. It has a score. It has tiers. It probably has a color scale. It gets added to a query, a portal, a counselor list, or a leadership report. Then someone asks what should happen when the score moves. The answer is vague.

Or there is a campaign. It is clever. It uses conditional content. It has good audience logic. It probably took a patient Slate person a long time to build. Then the owner leaves, and nobody knows where the logic lives, why the audience was defined that way, or whether the campaign is still doing what it was meant to do.

That pattern appeared again and again across the Slate Summit material.

The strongest work was about more than what Slate can do. It was about the institutional decision Slate is helping people make.

That is the central move in this chapter. Slate is no longer only a configuration system. At its best, it is decision infrastructure.

Decision infrastructure means the reports, rules, portals, queries, campaigns, datasets, scores, forms, and dashboards are built around a practical chain of use:

1. A decision the institution needs to make.
2. A person or group responsible for that decision.
3. Evidence that helps the decision get made.
4. An action that changes because of the evidence.
5. A review habit that shows whether the work is helping.

The phrase decision infrastructure is my synthesis. It comes from reading across several Summit sessions that were, on the surface, about different things: change management, Querybase Views, yield volatility, and origin source reporting.

The featured sessions are:

1. "Durable Change By Design: From Human Insight to Decision Infrastructure," presented by Ben Parsons and Sarah Kotlinski from College of the Holy Cross and Gettysburg College.
2. "What a View! Querybase Views for Quick Wins," presented by Mara Marzocchi from Suffolk University Graduate Admissions.

3. "Adapting to Yield Volatility: Using Data and Technology to Shape the Right Class," presented by Catie Clark, Christie Smith, Olivia Nash, and Ally Metcalf from Niche, Alfred University, University of the Pacific, and The University of Tulsa.
4. "From Rough Demo to Data Hit: Telling Your Origin Story," presented by Russell Goff, Ashley Valladares, and Kylie Hall from Sherman College of Chiropractic and enrollmentFUEL.

These sessions do different work. Together, they point toward the same operating question:

What institutional judgment is this build supposed to improve?

The Shared Pattern

"Durable Change By Design" is useful because it names something Slate teams experience often and discuss too rarely: change creates loss.

A new workflow can be better and still disrupt habits. A new process can be more durable and still threaten the confidence people had in the old informal way. A system can be cleaner and still make staff feel as if their judgment has been flattened into fields, rules, and reminders.

That is why the session matters. It treats human judgment, decision rationale, vendor context, contract details, and institutional memory as things worth preserving inside the system. Its examples use Organization Records, custom datasets, forms, materials, queries, dashboards, and review reminders to keep decisions from disappearing into personal notes, private messages, or the memory of whoever built the object.

The vendor and contract example is especially useful. The decision is recurring. Should the institution renew the vendor? Should the relationship change? Is the contract still serving its purpose? Who should review it, and when?

If the reasoning behind the original decision is gone, the next review starts from fog. That is more than a documentation problem. It is a decision problem.

"What a View! Querybase Views for Quick Wins" comes from a more technical angle. The session focuses on Querybase Views, with examples around expensive reports, guardrails for less technical users, integrations, Ping reporting, and shaping data for safer reuse.

Under the technical layer is the same question: how do we make the right evidence easier to use?

A Querybase View has value when it changes how people access, trust, reuse, or act on data. If a funnel report runs in two seconds instead of twenty, that speed can change whether staff actually use the report in the rhythm of work. If a view gives faculty or staff a safer population to query, the view is also a governance choice. If Ping data or UTM information is reshaped so it can be interpreted, the technical work makes digital behavior usable for a decision.

The build reduces friction between a question and evidence. That is decision infrastructure.

"Adapting to Yield Volatility" brings the leadership context. Enrollment outcomes are harder to predict. Application behavior, demographics, melt, competition, aid, policy, and student expectations are shifting at the same time.

The lesson I take from this session is that application volume alone is a weak strategy. Institutions need better ways to see fit, conversion, barriers, behavior, channel performance, and campus context while the cycle is still moving.

If an office only understands its yield story after May 1, it is doing archaeology. Sometimes that work is necessary. The value comes when the next cycle gets better.

Yield strategy needs infrastructure that helps the institution observe, interpret, and act while there is still time to change something.

"From Rough Demo to Data Hit" focuses on Origin Sources and Origin Groups. The Sherman College of Chiropractic example shows how messy source structures can weaken recruitment analysis.

The key discipline is simple: origin should answer the initial hand raise.

It should not be forced to explain the entire student journey.

That distinction matters because offices often ask source data to answer too many questions at once. Where did the student first enter? Which channel deserves credit? What influenced the application? Which campaign worked? Which event mattered? Which relationship helped?

Those are related questions. They are not the same question.

If one field or one taxonomy is expected to answer all of them, the data eventually starts answering a question no one meant to ask. The session offers a practical method: audit current sources, identify gaps, map record entry paths, rebuild source groups, refresh records, wait for recalculation, and repeat.

On the surface, that is source governance. Underneath, it is decision infrastructure. If origin data guides recruitment investment, travel planning, campaign strategy, or channel evaluation, the source structure has to be built for that decision.

Four Lessons

The four sessions differ in topic, toolset, and audience. They still agree on four lessons.

Evidence Has To Be Shaped

Raw data is rarely ready for strategy.

A slow report is not strategy. A source field with hundreds of values is not strategy. A predictive score without an action rule is not strategy. A contract folder no one reviews is not strategy.

Evidence becomes useful when it is structured around a decision.

That may mean precomputing expensive logic. It may mean limiting a population. It may mean translating field values. It may mean separating current reporting from historical reporting. It may mean cleaning source groups. It may mean deciding that origin answers the first hand raise rather than the full recruitment journey.

The practical question is simple: when a user opens this object, what evidence should they trust?

The answer should be narrower than all available data. It should be the evidence that matters for the decision.

Ownership Turns Output Into Practice

A Slate object can have a technical owner and a decision owner. These roles may sit with different people.

The technical owner builds the query, rule, dashboard, portal, view, or campaign. The decision owner is accountable for what happens because of it.

For a yield dashboard, the decision owner might be admissions leadership. For a counselor priority list, it might be a territory manager or director. For a financial aid barrier report, it might be shared by admissions and financial aid. For a vendor review dashboard, it might be operations leadership. For origin source governance, it might be enrollment analytics or admissions operations.

If no one owns the decision, the build can still exist. It will lack authority. Action becomes optional, and optional action tends to fade.

Speed Has To Serve Judgment

Speed matters. Faster reports, cleaner views, and earlier signals can help teams act while action is still useful.

The risk is confusing speed with quality.

Faster weak decisions are still weak decisions. The craft is building enough structure that people can act sooner without acting blindly.

Querybase Views can make evidence easier to reach. Yield analytics can help teams notice movement during the cycle. Origin source cleanup can make channel analysis more credible. Each one matters because it gives judgment a better surface to work from.

Memory Is Infrastructure

Institutional memory is part of the system.

People change jobs. Offices reorganize. Vendors change products. Consultants finish projects. Staff inherit Slate instances. New leaders ask old questions with new urgency.

If the system does not preserve reasoning, the institution has to relearn its own history. That cost is usually invisible until something breaks.

This is where the change management session and the origin source session connect strongly. One asks how decision reasoning survives change. The other asks how source history and analytic trust survive messy entry paths. Both are concerned with the same problem: what happens when the people who remember why this exists are no longer in the room?

The Five Question Test

Every meaningful Slate build should answer five questions.

1. What decision does this support?
2. Who owns that decision?
3. What evidence does the build surface?
4. What action should change because of it?
5. How will we know if it is working?

These questions sound basic. That is why they are useful. They force the build to explain itself.

What Decision Does This Support?

A report might help decide which admitted students counselors should prioritize this week. A dashboard might help decide which source channels deserve more investment. A portal might help students understand their next step. A vendor record might help leaders decide which contracts need review. A campaign might help determine which audience needs a different message.

The test is to finish one sentence:

"This build helps us decide whether..."

"This dashboard helps us decide which..."

"This score helps us decide when..."

"This portal helps students decide how..."

If the sentence cannot be finished, pause the build long enough to clarify the work.

Who Owns The Decision?

Ownership makes governance real.

If a build supports counselor outreach, who decides how counselors should use it? If a score creates action bands, who approves those bands? If source groups are being cleaned, who decides what the groups mean? If a dashboard is used in leadership meetings, who is responsible for the decision cadence?

Without a decision owner, the technical object may continue running while the practice around it slowly dissolves.

What Evidence Does The Build Surface?

This is where evidence design matters.

For counselor outreach, the relevant evidence may include recent event attendance, portal activity, aid status, application stage, and territory. For source investment, the evidence may include clean origin groups, inquiry to application conversion, application to admit conversion, admit to deposit conversion, and staff effort. For vendor review, the evidence may include contract dates, cost, owner, use case, assessment score, and upcoming review window.

If the evidence is limited, say that. A build that names uncertainty is more trustworthy than one that hides it.

What Action Should Change?

This is where many builds fail. They produce information. They do not define use.

A dashboard should clarify what people should consider doing next. A score should define action bands. A campaign should move a student toward a useful next step. A Querybase View should make a process safer, faster, or more consistent.

Examples help.

If admitted students attended a high intent event and have not deposited, the action might be counselor follow up within three business days.

If a student has high academic fit and low engagement, the action might be a different communication path.

If a vendor contract is within 90 days of renewal and has no current evaluation, the action might be leadership review.

If a source group has too many unmapped values, the action might be source governance before the next campaign analysis.

A bar of iron becomes a tool when it has a shape, a handle, an edge, and a purpose. A report works the same way.

How Will We Know If It Is Working?

The answer does not have to be a formal experiment. It does need some way to know whether the object is useful.

For a dashboard, the measures might be usage, meeting cadence, reduced manual reporting, and clearer priorities. For a campaign, they might be response, appointments, movement, deposits, or reduced confusion. For a Querybase View, they might be performance improvement, fewer query errors, safer access, or cleaner integrations. For origin source cleanup, they might be fewer unmapped sources and greater confidence in channel analysis. For decision governance, they might be fewer missed renewals, better onboarding, and less reliance on individual memory.

The key is to avoid overclaiming. Operational evidence, promising patterns, and verified causal impact are different things. Mature decision infrastructure keeps those categories separate.

Applying The Framework

Start with one build.

Choose a dashboard, campaign, model, report, portal, event follow up process, vendor review process, or origin source structure. Pick something people use or should use.

Write the decision sentence:

"This build helps us decide..."

Then name two owners if needed: the technical owner and the decision owner.

Next, define the action rule. It can be simple.

If this population appears, a counselor reviews it every Monday.

If this contract is within 90 days of renewal, leadership reviews it.

If this source is unmapped, it enters a monthly source governance queue.

If this signal is high and deposit is missing, the student receives a specific follow up.

If this report loads too slowly, the expensive logic moves into a scheduled structure.

Finally, add a review date. Add one for dashboards, campaigns, Querybase Views, source mappings, predictive models, portals, and rules. A build without a review date becomes a future mystery.

How To Study The Featured Sessions

Study "Durable Change By Design" for the way decision rationale becomes structured data. Notice how Organization Records, datasets, forms, dashboards, contract documents, review windows, and decision rubrics preserve reasoning that would otherwise drift into email and memory.

Study "What a View!" for architecture criteria. Ask when data is expensive to calculate repeatedly, when a population needs guardrails, when scheduled lag is acceptable, when a view is temporary, and whether the build reduces risk or only moves complexity somewhere else.

Study "Adapting to Yield Volatility" for posture. Yield volatility requires staged testing, connected data, behavioral interpretation, and a willingness to challenge volume based assumptions.

Study "From Rough Demo to Data Hit" for analytic discipline. Origin data should answer a specific question. If an office wants a different answer, it may need a different structure.

That lesson applies far beyond Origin Sources. It applies to every Slate field that slowly becomes overloaded with meaning.

Closing

Advanced Slate work is becoming decision infrastructure.

That does not mean every build has to be large. Some of the best infrastructure is small: a clean source taxonomy, a fast view, a clear campaign map, a review reminder, a dashboard with five useful numbers, a score with an action policy, or a form that captures decision rationale before it disappears.

The craft is not making Slate do more. The craft is making Slate help the institution decide better.

So pick one build and write the decision sentence.

Then name the owner, define the evidence, connect it to action, and set a review date.

That is where a Slate object starts becoming infrastructure.

Source Notes

1. S20, "Durable Change By Design: From Human Insight to Decision Infrastructure," Ben Parsons and Sarah Kotlinski, College of the Holy Cross and Gettysburg College. Key source pages include 7 to 17 for change and grief lenses, 20 to 23 for the Three Gates and blueprint, and 25 to 35 for vendor records, contracts, dashboards, 90 day query, and annual review.
2. S88, "What a View! Querybase Views for Quick Wins," Mara Marzocchi, Suffolk University Graduate Admissions. Key source pages include 5 to 15 for creation, exports, filters, translations,

and privacy, 17 to 28 for funnel improvement, 29 to 32 for population guardrails, 33 to 41 for Ping and Voyager reporting, and 53 for architecture criteria.
3. S03, "Adapting to Yield Volatility: Using Data and Technology to Shape the Right Class," Catie Clark, Christie Smith, Olivia Nash, and Ally Metcalf, Niche, Alfred University, University of the Pacific, and The University of Tulsa. Key source pages include 5 to 10 for market changes and strategy, 12 to 18 for Alfred measures and targeting, 20 to 29 for Pacific integrations, AI analysis, portal personalization, and melt strategy, and 31 to 33 for Tulsa direct admission growth.
4. S33, "From Rough Demo to Data Hit: Telling Your Origin Story," Russell Goff, Ashley Valladares, and Kylie Hall, Sherman College of Chiropractic and enrollmentFUEL. Key source pages include 7 to 14 for the problem and five step method, 20 for the audit query, 21 to 32 for mapping, source creation, and gap reports, 34 to 39 for lessons and recalculation, and 41 for the final framework.

Chapter 2 - The Query Ladder

There is a moment in Slate work that tells you almost everything about an office.

Someone asks what sounds like a simple question.

How many admitted students from New Jersey attended an event and still have no deposit?

Which applicants are missing a material, only for programs where that material is actually required?

How many students first entered through this source, and how many of them eventually enrolled?

The question sounds straightforward. Then the room pauses, because everyone knows the answer depends on who builds the query.

One person starts on person. Another starts on application. Someone filters on rank one address and misses the address they needed. Someone exports all addresses while filtering on one address and cannot explain why the result changed. Someone says the office should use a subquery. Someone else asks why. Then the one Slate person who understands Configurable Joins becomes the whole reporting strategy.

That is the problem this chapter is about.

Queries are technical. The bigger issue is that query knowledge is often concentrated in too few people, while the institution depends on the results. If a team depends on one person to understand joins, the office has a bottleneck.

The Query Ladder is a way to name the learning path. It asks what level of query literacy Slate professionals need for the decisions they support.

Different roles need different levels of query depth. More people need enough query literacy to understand what question they are asking, what one row represents, where the data lives, and when the output might mislead them.

This chapter draws from five Slate Summit sessions:

1. "Fantastic Data & Where to Find It: Demystifying Tables, Joins, and Relationships," presented by Kristie Rickerd from Linfield University and Claire Hirte from enrollmentFUEL.
2. "Yes, You Need to Know Configurable Joins: A Beginner's Guide to Configurable Joins," presented by Danielle Buczek and Paula Schaefer Riley from Post Captain Consulting and Laura Sturgill from West Chester University Graduate Admissions.
3. "A Join Named Sue," presented by Charles B. Wilson from The University of Chicago.
4. "But Why Subqueries," presented by Jason Davitt from University of Scranton Graduate Admissions.

5. "Technolutions Training: Query Grab Bag, An Assortment of Next Level Querying Techniques," presented by Louise Karneus and Max Zeryck from Technolutions.

Those sessions move from beginner mental models to advanced query techniques. The useful synthesis is a learning path. Query literacy is now part of the craft base for modern Slate work.

Why Query Literacy Matters

A Slate query is rarely just a query.

It decides who gets an email, who appears on a counselor list, who is counted in a dashboard, who triggers a rule, who is included in an export, which students are considered engaged, and which records appear to be missing something important.

Query logic becomes institutional logic.

If the query is wrong, the institution may send the wrong message, miss the wrong students, overcount the wrong source, undercount the wrong population, misunderstand the funnel, or make a decision with unearned confidence.

That is why query literacy matters. Query logic determines what the institution sees, and what the institution sees shapes what it does.

The beginner sessions matter because they start where the work starts.

"Fantastic Data & Where to Find It" grounds the learner in Slate's data world: where data lives, what tables are, how records relate to each other, and why relationships matter. The practical lesson is straightforward: choose the correct base, understand one to many relationships, prefer direct paths where possible, use the Almanac, label joins clearly, and build queries another person can understand.

"Yes, You Need to Know Configurable Joins" reinforces the concept every Slate office should teach early: row grain.

Row grain answers one question: what should one row represent?

One person? One application? One decision? One address? One event registration? One interaction? One material?

If the team cannot answer that before building, the query is already in trouble.

Many messy reports come from skipping this step. A team asks a person level question and starts on applications. It asks an application level question and exports person level fields. It asks whether someone has an address in one state and then tries to export every address. It asks how many students attended an event and accidentally counts registrations, event instances, or duplicate rows.

That is more than a technical mistake. It is a decision mistake.

The row grain defines what the institution thinks it is counting.

The Ladder

The Query Ladder has six rungs:

1. The question.
2. Row grain.
3. Bases and direct paths.
4. Many related rows.
5. Advanced comparisons.
6. Reusable assets.

Each rung adds a form of judgment. The ladder is about clarity. A counselor, communication builder, analyst, director, and advanced Slate developer may need different depth. They still benefit from a shared vocabulary.

Rung 1: The Question

The first move is to slow the question down.

"How many admitted students attended an event and still have no deposit?" sounds clear. The query has to answer several hidden questions.

Does admitted mean any admit decision, latest decision, active application, or current cycle only?

Does attended mean registration, check in, attendance status, guest attendance, or participation in a specific event population?

Does deposit mean payment, checklist completion, active enrollment intention, or a current application status?

Should the answer count people or applications?

Those choices are part of the analytic work. If the office cannot name them, the query builder has to guess. When the query builder guesses, the institution may treat a private assumption as a shared definition.

A better starting habit is to write the question as a definition before building the query.

For example:

This query identifies one person per admitted student in the current first year cycle who attended at least one admitted student event and has no active deposit on their current application.

That sentence will never fit every use case. That is the point. It forces the team to see the assumptions before they become output.

Rung 2: Row Grain

The first real move on the Query Ladder is defining what one row means before choosing a join.

If one row represents a person, the query should usually support a person level action: send one email, create one counselor list row, count one student.

If one row represents an application, the query can support application level work: checklist requirements, program specific decisions, application status, or review movement.

If one row represents an event registration, the query can analyze attendance behavior, registrations, no shows, and event capacity.

If one row represents a decision, the query can analyze decision patterns rather than people.

The base should match the row grain.

Person based queries are useful when the action happens once per student. Application based queries are useful when each application needs separate treatment. Decision based queries are useful when the decision itself is the unit of analysis. Event registration based queries are useful when the event record is the unit being counted.

This is where many offices drift. A query starts as one thing, then new exports and filters are added over time. After enough copying, no one can say what one row represents anymore.

The fix is basic and powerful: add a row grain sentence to important queries.

This query is person based because the communication should send once per person.

This query is application based because the requirement is evaluated separately for each application.

This query is decision based because we are analyzing decisions rather than students.

This query is event registration based because we are analyzing attendance records.

That sentence can live in the query description, a documentation page, a training guide, or a trusted query library. It prevents future confusion.

Rung 3: Bases And Direct Paths

Once the row grain is clear, the next move is choosing the right base and relationship path.

Slate is full of one to many relationships. One person can have many applications. One application can have many materials. One person can have many addresses. One event can have many registrations. One school can connect to many students. One person can have many interactions.

If the base is wrong, rows can multiply, collapse, disappear, or get selected from many possible options in ways the user cannot explain.

This is where the advice from "Fantastic Data" becomes practical. Choose direct paths where possible. Use the Almanac. Label joins clearly. Build something another person can inherit.

A join label is a note to future staff. It says the relationship was intentional.

Use a clearer label than "school" when the real meaning is "rank one school for current application review."

Use a clearer label than "address" when the real meaning is "current mailing address for communication audience."

Use a clearer label than "event" when the real meaning is "attended admitted student event in current cycle."

Labels should explain why the relationship exists. That habit turns a query from private knowledge into shared work.

Rung 4: Many Related Rows

Many related rows are the point where a lot of query confusion starts.

If the base row is one person and that person has many addresses, the query has to decide what to do with those addresses. Choose one. Count them. Test whether any exist. Concatenate them. Aggregate them. Filter based on them while keeping the person row intact.

That is why Jason Davitt's "But Why Subqueries" is so useful. The title captures the learner's real frustration. The practical reason is simple: one to many data behaves differently from one flat table.

Subqueries help inspect related rows while preserving the base row of the main query.

A subquery can ask whether a person has any address in a region. It can count applications. It can test whether an application has missing required materials. It can identify the most recent event attended. It can check whether a student has an interaction matching a specific type.

The goal is pattern recognition rather than universal subquery expertise.

I am asking a question about many related rows. I need a structure designed for many rows.

That recognition changes the work. It helps users stop forcing complex data into a simple row expectation.

Rung 5: Advanced Comparisons

Charles B. Wilson's "A Join Named Sue" moves into advanced territory. The session handles situations where Slate lacks a native relationship that matches the business process.

The session teaches a way to create a business defined relationship using independent subqueries and comparison exports. The session calls this a Sue join.

The strategic idea matters even for people who never build one.

Sometimes the relationship an institution needs lacks a native relationship in Slate. The institution knows two things should be compared. Slate may have no built in join that says they belong together. The advanced builder creates a controlled comparison.

The examples include a guest to host self join for an overnight event and a program management dataset joined to applications by program, round, and track. The reusable logic can support supplements, checklist groups, application pages, failures, bins, feeds, reports, and status portal content.

That kind of work requires governance.

When advanced builders create business defined joins, they are encoding institutional logic. That logic needs naming standards, documentation, null handling, testing, regression checks, and a clear owner. Someone has to understand what happens when a program changes, a round changes, a track changes, or a dataset value is missing.

The lesson is precise: advanced query techniques can become governance tools when used well, and future mysteries when used casually.

Rung 6: Reusable Assets

The Technolutions "Query Grab Bag" session adds the final layer: reuse.

The session covers independent subqueries, dictionary exports, query based reporting, and parameters. The common thread is durability.

How can a team avoid rebuilding the same logic every year?

How can multi row outputs be structured so Liquid can use them?

How can parameters keep query logic adaptable without hardcoding cycle specific values?

How can comparisons work when the base relationship is complex?

This is where query work starts to become platform architecture.

A dictionary export can become structured material for Liquid and interfaces. A parameter can make a query adaptable across cycles. A reusable query pattern can reduce the chance that five people build five different definitions of the same idea.

If every campaign rebuilds its own version of high engagement, the office lacks a signal strategy. If every report defines demonstrated interest differently, the office lacks an analytics strategy. If every year specific query hardcodes the cycle, the office lacks durable infrastructure.

Reusable query assets turn query skill into institutional infrastructure.

Query Literacy By Role

A strong Slate office should have a shared vocabulary for query logic. Roles need different depths.

A counselor may need to understand the row grain of the lists and reports they use. They should know whether a list is one row per person, one row per application, or one row per event record.

A communications builder may need to understand populations, exclusions, joins, one to many risk, and timing. If the query drives a message, the query defines the audience.

An analyst may need to understand bases, aggregates, subqueries, source definitions, deduping, and validation. If the query feeds a dashboard or model, its assumptions become part of the analysis.

An advanced Slate developer may need independent subqueries, comparison exports, dictionaries, parameters, query libraries, performance testing, and governance practices.

The shared vocabulary matters because Slate work crosses roles. A director should be able to ask what one row represents. A counselor should be able to ask who is excluded. A communication owner should be able to ask which event behavior counts. An analyst should be able to ask whether a definition changed across cycles.

Those questions make the work safer.

Three Applied Examples

Consider an admitted student event campaign.

The communications team wants to send a message to admitted students who attended an admitted student event and have no deposit. The query has to decide whether the communication

is person based or application based. It has to define attendance. It has to decide whether all admitted student events count or only selected event populations. It has to know whether the event record is tied to the person while the deposit is tied to the application. It has to account for students who deposit after the query runs.

This is communications governance expressed through query logic.

Consider a counselor priority list.

Leadership wants a weekly list of admitted students with high engagement and no deposit. The query has to define engagement. Portal login, event attendance, email clicks, Ping activity, interview, FAFSA, parent engagement, and counselor interaction are different signals. The query also has to decide the row grain. One student per counselor? One application? One student per major? What happens if the student has multiple interests, territories, or event records?

The counselor can work without every technical detail. Someone still needs to be able to explain the logic.

Consider predictive modeling.

A model is only as strong as the features feeding it. If a feature says attended event, the model inherits whatever that query means. Any event ever? Recruitment event? Post admit event? In person only? Virtual? Student attended? Guest attended? Registered and absent? If a feature says email click, which clicks count? All clicks, high intent links, parent clicks, student clicks, pre admit clicks, post admit clicks?

The model may be built outside Slate. The raw material often comes from Slate. If the query logic is unstable, the model inherits that instability.

The work starts before the model. It starts when the team decides what the data means.

Governance And Access

Query literacy should grow with governance.

Some bases are sensitive. Some data should have tight permissions. Some users should work through safer views, shared reports, or defined populations. Supporting more query literacy works best with guardrails around access.

The better move is to pair literacy with guardrails.

Less technical users can learn what row grain means while still using approved reports. Communications teams can understand audience logic while working from trusted patterns. Analysts can document reusable definitions. Advanced builders can create complex assets with review, testing, and ownership.

Advanced query techniques can also create technical debt. Independent subqueries, comparison logic, dictionary exports, parameters, and reusable libraries are powerful. Power without documentation is a future problem.

If one person understands a complex query, document it.

If a query drives rules, checklists, portals, bins, communications, reports, or models, test it.

If a parameter changes each year, assign an owner.

If a dictionary export feeds Liquid, document both the query and the rendering logic.

That is how query skill stays useful after the original builder moves on.

Applying The Framework

Start by building a Query Ladder for your office.

At the bottom, teach record bases and row grain. Then teach direct joins and one to one relationships. Then teach one to many relationships and why duplicates happen. Then teach subquery filters and exports. Then teach aggregates, ranking, and testing. Then teach independent subqueries and business defined comparisons. Then teach reusable assets: query libraries, dictionaries, parameters, materialized views, and documentation.

Next, add a row grain sentence to important query documentation.

One row represents a person.

One row represents an application.

One row represents a decision.

One row represents an event registration.

Then add why the base was chosen.

Next, label joins as if a future coworker will inherit the query without context. The label should explain the relationship, the selection, and the reason it exists.

Next, build a query review habit. Before a query drives a major communication, dashboard, rule, model, export, or leadership report, review the base, row grain, many row relationships, exclusions, sensitive data exposure, and places where Slate selects one row from many.

This can be a fifteen minute peer review rather than a formal committee.

Finally, create a trusted query pattern library.

Every institution has patterns that recur: admitted and no deposit, registered and absent, attended a high intent event, complete application with missing financial aid step, high engagement and no response, students with any address in a region, students whose latest decision meets a condition.

Document the base. Document the row grain. Document the logic. Document when to avoid the pattern.

A trusted pattern should include warning notes:

This pattern is person based. Avoid it when you need one row per application.

This pattern checks any address. Avoid it when you need current mailing address only.

This pattern identifies attendance. Avoid it when you need registration.

That is how query skill becomes institutional infrastructure.

How To Study The Featured Sessions

Study "Fantastic Data & Where to Find It" first. Use it to ground newer users in tables, bases, joins, relationships, ranking, and efficient query habits.

Study "Yes, You Need to Know Configurable Joins" next. Use it to teach vocabulary, relational thinking, row grain, direct joins, main level joins, and subquery joins.

Study "But Why Subqueries" after that. Use it to explain why one to many data creates confusion and when subqueries become necessary.

Study "A Join Named Sue" with advanced builders. Use it to understand business defined relationships, controlled comparison logic, independent subqueries, and reusable logic that can support multiple Slate objects.

Study "Query Grab Bag" once the team is ready for reusable techniques. Use it for independent subqueries, dictionaries, query based reporting, parameters, and durable query patterns.

The path moves from understanding the data world, to asking better questions, to choosing better bases, to handling many related rows, to creating reusable advanced logic.

That is the Query Ladder.

Closing

Query literacy is now part of the professional craft of Slate.

Institutions make decisions based on query output every day. Who gets a message. Who appears on a list. Who is counted in a funnel. Who triggers a workflow. Who is considered engaged. Who is flagged for review. Who is missing from the report entirely.

Query logic is institutional logic.

Institutional logic belongs in shared practice, documentation, and training.

Pick one important query your office uses. Open it. Leave the logic unchanged for now. Write down what one row represents, why the base was chosen, where the query touches many related rows, and whether another person could explain it six months from now.

If another person would struggle to explain it, you found your next training artifact.

The goal is to make querying less mysterious and more teachable.

Start with one query. Write the row grain sentence.

Source Notes

1. S26, "Fantastic Data & Where to Find It: Demystifying Tables, Joins, and Relationships," Kristie Rickerd and Claire Hirte, Linfield University and enrollmentFUEL. Key source pages include 7 to 16 for records, tables, bases, joins, and ranking, 17 to 21 for query efficiency practices, and 22 for knowledge check.
2. S92, "Yes, You Need to Know Configurable Joins: A Beginner's Guide to Configurable Joins," Danielle Buczek, Laura Sturgill, and Paula Schaefer Riley, Post Captain Consulting and West Chester University Graduate Admissions. Key source pages include 5 to 17 for terminology, relational model, query interface, and join types, 21 to 27 for one to one and rank examples, 30 to 37 for main level versus subquery joins and address examples, and 38 to 40 for Configurable Joins across Slate and the companion portal.
3. S01, "A Join Named Sue," Charles B. Wilson, The University of Chicago. Key source pages include 15 to 17 for Configurable Joins explorer, 22 to 67 for bases and standard joins, 68 to 84 for subquery exports and filters, 89 to 136 for guest to host self join, 142 to 179 for program dataset join, and 181 to 211 for reusable rules, logic, checklists, feeds, and portal uses.
4. S14, "But Why Subqueries," Jason Davitt, University of Scranton Graduate Admissions. Key source pages include 8 to 17 for bases and joins and 18 to 59 for subquery patterns and examples.
5. S76, "Technolutions Training: Query Grab Bag, An Assortment of Next Level Querying Techniques," Louise Karneus and Max Zeryck, Technolutions. Key source pages include 5 to 25 for independent subqueries, 27 to 40 for dictionaries and XML, 42 to 48 for reporting with queries, and 50 to 64 for parameters and reusable query examples.

Chapter 3 - From Signals to Action

There is a moment in almost every Slate shop when someone looks at a student record and says, "This student is clearly interested."

Then the harder question arrives.

Interested enough for what?

That is where the work gets real. Slate can collect signals. Portals, forms, events, Deliver, Ping, source formats, vendor platforms, interactions, and community tools can produce a lot of behavioral data. The harder question is whether the institution knows when a signal becomes actionable.

A click is weak strategy. A score is weak personalization. A model is weak permission. A dashboard is weak handoff.

A useful signal needs an action policy. It should answer what the behavior probably means, how much confidence the team has, who may act on it, what action is appropriate, what should be suppressed, and how the team will know whether the action helped.

The raw material is student behavior. The finished tool is a disciplined decision rule that staff can use without guessing.

This chapter draws mainly from four Slate Summit sessions:

1. "Student Signals, Stranger Things, and the Path to Personalized Outreach," presented by Moni Nann and Ben Parsons from College of the Holy Cross, with Courtney Buckner and Katy Remich from ZeeMee.
2. "From Insight to Action: Using Behavioral Data to Power Targeted Campaigns," presented by Aimee Celentano and Amanda Marsili from Roger Williams University Undergraduate Admissions, with Megan Gabbard from Capture Higher Ed.
3. "All the Small Clicks: Conversion Rate Optimization Lessons from Ads to Applications," presented by Lillie Miller, Michael Doran, and Launa Smith from the University of Georgia Undergraduate Admissions.
4. "Adapting to Yield Volatility: Using Data and Technology to Shape the Right Class," presented by Catie Clark, Christie Smith, Olivia Nash, and Ally Metcalf from Niche, Alfred University, University of the Pacific, and The University of Tulsa.

Together, these sessions point toward a clear shift.

Slate teams are moving from reporting what happened to deciding what should happen next.

That shift requires more than data collection. It requires governed behavioral data that leads to defined, proportionate, measurable action.

The word proportionate matters.

Some signals deserve a counselor task. Some deserve a campaign change. Some deserve a portal adjustment. Some deserve staff awareness. Some deserve suppression. Some deserve research first. Some deserve no action until they repeat or combine with another signal.

The question is simple:

When does engagement or behavioral data become actionable enough to change outreach?

Why Signals Need Governance

Student behavior creates evidence. It rarely creates instructions by itself.

A student sends an email. That is a signal. A student visits a financial aid page several times. That is a signal. A student registers for an admitted student event and misses it. That is a signal. A parent clicks scholarship content. That may be a signal. A student joins a community platform, logs into the portal repeatedly, starts a form, leaves, returns two days later, and then goes quiet again. That is a set of signals.

None of that automatically tells the office what to do.

This is where institutions get stuck. They collect more fields, build more tabs, add more dashboards, and assume action will follow. Usually action still depends on a person interpreting the record from scratch.

The counselor decides whether the student deserves outreach. The communications team decides which campaign fits. The events team decides whether another reminder helps or adds noise. Leadership decides whether a segment is truly at risk or merely quiet.

When those decisions remain undefined, the office has signal clutter instead of a signal system.

A signal system has three parts:

1. A behavior the institution can observe with enough reliability.
2. An interpretation that is cautious, clear, and limited.
3. An allowed action with an owner, timing, suppressions, and measurement.

The third part is where many systems are weakest.

The Holy Cross and ZeeMee session is useful because it shows a signal to action pattern across websites, events, portals, forms, email, and ZeeMee. The important move is the progression from signal definition to Slate structure to activation through portals, campaigns, notifications, counselor queries, and leadership views.

The architecture is easy to state and hard to practice:

Define the signal. Structure it. Decide how it may be used.

That last step is an operating decision. If one signal can affect a portal, an email, a counselor notification, and a leadership dashboard, the institution has to decide which uses are appropriate.

A student who shows strong community engagement may be a good candidate for a student life message. Should that same signal affect counselor priority? Should it influence scholarship outreach? Should it appear in a leadership yield view? Should every user see it? Should it expire after thirty days?

Those are governance questions. Slate can support the action once the institution decides the policy.

Signals By Funnel Decision

The Roger Williams and Capture session gives the operational version of the same problem. Their presentation describes a two way data exchange: Slate sends data out through daily queries and SFTP, behavioral fields and model outputs come back through source formats, and the returned data becomes visible through record tabs, popovers, reports, query filters, counselor workflows, and campaign actions.

That is practical machinery. It also increases the governance burden.

One useful part of the Roger Williams model is the separation of apply, yield, and anti melt use cases. That distinction matters because the same behavior can mean different things at different stages.

Before application, a signal might suggest readiness to apply. After admission, it might suggest yield interest, uncertainty, or price concern. After deposit, it might suggest onboarding friction or melt risk.

One generic engagement score flattens those differences.

The better pattern is to build Signal Indices around decisions.

An application generation signal should answer: who needs help moving from interest to application?

A yield signal should answer: who is showing commitment, comparison, hesitation, or price concern?

An anti melt signal should answer: who has deposited and may be stuck, delayed, or drifting?

Those are different questions. They need different inputs, thresholds, actions, owners, and success measures.

A Signal Index should authorize a specific kind of action. If the index cannot tell the team what should change, it is closer to decoration than infrastructure.

This is where the language of signals has to become more precise.

High engagement is too broad for most action. High admitted student event engagement with no deposit is clearer. Recent financial aid page engagement with no aid appointment is clearer. Parent engagement present with limited student engagement is clearer. Deposited student missing onboarding step is clearer.

The signal label should describe the observed behavior before it assigns meaning.

That discipline protects staff from overreading the record.

Experience Design Creates Signal Quality

The University of Georgia session adds a layer that is easy to understate: signal quality starts with experience design.

The session is about conversion rate optimization from ads to applications. It covers portals and forms as advertising collateral, tracking through tools like GTM, platform pixels, Slate Ping, UTM tracking, and Common App TREFs, and reporting that follows students from RFI activity through application and deposit.

The technical pieces matter. The deeper lesson is that a messy student path creates messy signal interpretation.

If a landing page has too many calls to action, a click tells the team less. If an RFI form asks for too much too soon, abandonment may mean frustration rather than low interest. If metadata is inconsistent, attribution weakens. If the mobile experience is poor, student behavior may reflect design failure instead of student intent.

Conversion rate optimization and signal strategy belong together.

Every form field is a future data point. Every call to action is a future interpretation problem. Every tracking parameter is a future reporting dependency. Every portal decision changes what behavior means.

Teams that want better engagement scoring should start before the scoring formula. Make the path cleaner. Make the choice clearer. Make the data more consistent. Then score the signal.

This has practical consequences.

An inquiry form that asks one clear question about academic interest can support different follow up than a generic form with unclear intent. A portal that separates aid, housing, events, and

academic content gives cleaner behavioral context than one page with every link in one list. A landing page with one primary action produces a clearer signal than a page with six competing actions.

Better student experience creates better evidence.

Yield Needs Measured Precision

The yield volatility session gives the strategic reason this matters.

Catie Clark, Christie Smith, Olivia Nash, and Ally Metcalf are working from a market reality where application behavior, melt, financial aid, demographics, competition, and student expectations are shifting at the same time. In that environment, application volume alone is weak strategy.

The panel points toward staged direct admission, market groups, campus integrations, financial aid and housing signals, personalized admitted student portals, and AI assisted analysis of intent and melt profiles. The shared direction is admission as an ongoing signal environment.

A student is admitted, then the institution watches for evidence of commitment, hesitation, friction, readiness, and risk.

Watching still is not enough.

If a student has not completed housing, who acts? If a student has aid questions, what message is sent? If a student has high portal activity and no deposit, does that suggest interest or confusion? If a student appears at risk for melt, does the next action sit with admissions, financial aid, student affairs, or the academic college?

Without ownership, the signal sits there.

This is one of the most common failures in advanced Slate work. The institution creates a good report, and the report becomes another place where responsibility is vague.

A dashboard cannot own action. A query cannot own action. A score cannot own action.

If the action requires human judgment, name the role. If the action is automated, name the team responsible for the automation. If the action crosses offices, name the handoff.

The funnel has become too complex for simple stage based messaging. Inquiry, applicant, admit, and deposit still matter. They do not explain enough on their own.

Two admitted students may both be admits. One has visited campus twice, clicked financial aid content, opened housing information, and logged into the portal three times this week. Another has been quiet for a month. A third has parent engagement with limited student engagement. A fourth is active mainly around scholarship content. A fifth has deposited and has not completed onboarding steps.

Those students should receive different treatment only when the team can explain the difference, govern it, and measure whether it helped.

The right answer is governed differentiation.

Different enough to be useful. Governed enough to be trusted.

The Signal To Action Matrix

The practical tool for this work is a Signal To Action Matrix.

It does not need to be complicated. For every active signal or signal combination, define:

1. The observed behavior.
2. The possible interpretation.
3. The confidence level.
4. The allowed action.
5. The owner.
6. The timing and channel.
7. The suppressions.
8. The measurement plan.

The whole policy can be written as one sentence:

When this happens, we believe it may mean this, so this owner may take this action through this channel within this timeframe, unless these suppressions apply, and we will judge success by this measure.

If the team cannot finish that sentence, the signal is not ready to drive outreach.

It may belong in research. It may belong in a dashboard. It may be useful for staff awareness. It may need a pilot. It is not yet ready to change student treatment.

That is the difference between a signal and a policy.

Here is how the matrix changes a deposit campaign.

The blunt segment is admitted and not deposited. That population is a start. It is too broad for serious action.

A signal informed approach can separate the population into action groups:

1. High activity, deposit incomplete.
2. Financial aid engaged, deposit incomplete.
3. Event engaged, still undecided.

4. Portal quiet and email quiet.
5. Parent engagement present with limited student engagement.
6. Deposited, missing onboarding step.

Each group should have an owner, a message strategy, a suppression rule, and a success measure.

High activity with no deposit may call for clarity, reassurance, or a final counselor nudge. Financial aid engagement may call for cost clarity or an aid appointment. Event engagement may call for follow up tied to the event experience. Portal quiet and email quiet may call for a different channel or lower priority. Parent only engagement should not be treated as if the student personally clicked everything. Deposited and missing onboarding step may need a transition owner rather than an admissions message.

The point is controlled action groups. Endless microsegments create noise. Clear action groups create work the team can actually perform.

For a Funnel Insights Engine, this is where reporting becomes operational.

A basic funnel dashboard shows counts by stage. A more mature engine shows which stage specific behaviors are changing, which action groups are growing, which interventions are active, and whether those interventions are improving movement.

That is the shift from descriptive analytics to adaptive enrollment management.

Five Failure Modes

Signal systems fail in predictable ways.

Treating Engagement As Intent

Engagement and intent overlap. They are still different.

A student may be highly engaged because they are excited. They may also be confused, anxious, price sensitive, or comparison shopping. A quiet student may be disengaged, or already committed. Parent engagement may matter, and it is still different from student engagement.

Behavior is evidence. It is not confession.

Internal labels should reflect that humility. Be careful with labels like hot, cold, likely to enroll, or not interested unless the model behind them is strong and the action is governed.

Behavioral labels are safer:

1. Recent financial aid engagement.
2. High admitted student portal activity.

3. Event engaged and deposit incomplete.
4. Parent engagement present with limited student engagement.
5. Deposited student missing onboarding task.

Those labels tell staff what happened before they imply why it happened.

Crossing The Weirdness Line

Personalization works best when it connects to something the student knowingly did.

"You registered for our engineering webinar, so here are related resources" feels normal.

"We noticed you have been looking at scholarship content late at night" feels strange.

The test is simple: could a staff member explain the outreach to a student in one sentence without making the institution sound invasive?

If the answer is yes, the signal may be usable. If the answer is no, rewrite the action or keep the signal out of outreach.

False Precision

Scores compress messy behavior into a number. That can help staff prioritize. It can also create unearned confidence.

A student with an engagement score of 82 may not deserve meaningfully different treatment from a student with a 78. If the threshold is arbitrary, call it a pilot. If the score is a ranking tool, call it a ranking tool. If the model is vendor provided, make sure staff know what the label means and what it does not mean.

Focused indices are easier to defend than one master score.

A financial aid concern index is interpretable. A campus visit engagement index is interpretable. A deposit readiness index can be made interpretable. A broad student interest score may be harder to defend unless the definition, inputs, validation, and allowed actions are clear.

Message Fatigue

When every team acts on every signal, students get hammered.

A financial aid signal triggers one message. An event signal triggers another. A portal signal triggers another. A counselor query triggers outreach. A vendor segment triggers a campaign. The institution personalizes itself into noise.

Sometimes the best action is better sequencing. Sometimes it is better portal content. Sometimes it is counselor context. Sometimes it is suppression.

The strongest signal systems decide when to act and when to hold back.

Weak Evaluation

A signal action needs a success measure beyond opens and clicks.

For application generation, the measure may be application start, submission, or completion. For yield, it may be event attendance, aid appointment completion, deposit, or portal checklist progress. For anti melt, it may be housing completion, orientation registration, financial clearance, or reduced melt.

Click rate is not enrollment strategy.

The Georgia conversion rate optimization session matters because it connects upstream behavior with downstream application and deposit reporting. The work should follow the outcome that matters.

Staff Enablement

A signal system that staff do not understand will be ignored or misused.

Counselors need to know what a signal means and what it does not mean. Communications staff need to know which signals can change message logic and which should only inform reporting. Operations staff need to know how signals are sourced, refreshed, matched, expired, and audited. Leadership needs to know what claims can and cannot be made from the data.

If a counselor sees high engagement, they should not have to guess whether that means email clicks, portal activity, event attendance, vendor behavior, or all of the above.

If leadership sees high yield potential, they should know whether that is a validated model, a rules based index, or an experimental segment.

This is documentation work. It is also trust work.

A weekly counselor list becomes teachable when the logic is clear:

These students are admitted, have not deposited, have recent meaningful engagement, have no counselor interaction in the last seven days, and the recommended action is a short fit focused follow up. Do not mention tracking behavior. Log the interaction. Suppress if the student deposits or opts out.

That is much stronger than "call students with high engagement scores."

High compared to what? Based on what? For what purpose? With what message? How often? Who owns it? What if the parent caused the engagement? What if the student already said they are enrolling elsewhere?

A score is not an instruction. A query is not a strategy. A dashboard is not a handoff.

Staff enablement has to be part of the build.

AI And Signal Governance

The yield volatility session points toward AI assisted analysis of intent and melt profiles. Many institutions will experiment there, and they should. The same governance standard applies.

AI generated insight is not an allowed action by itself. It may suggest a pattern, summarize behavior, or help staff notice a segment. Before it changes outreach, someone still needs to define the action policy.

The review questions are direct:

1. What data did the AI use?
2. What action is being recommended?
3. Who reviews the recommendation?
4. Can staff explain it?
5. Can the student facing message stand on its own?
6. How will the team know if it helped?

AI does not remove governance. It increases the need for it.

Public conference sessions are best treated as patterns, not proof. A pattern says: here is a promising way to structure the work. Local implementation asks whether it works here, for these students, with this data, under these constraints, toward these goals.

That is the correct posture.

Applying The Framework

Start small.

Pick one decision where behavioral data should improve action.

Which admitted students should receive a counselor follow up this week?

Which non deposited admitted students need financial aid clarification?

Which application starters should receive completion support?

Which deposited students show signs of onboarding friction?

Choose one. Then build the Signal To Action Matrix for that decision.

List the signals the team can observe reliably in Slate: events, forms, portal activity, Deliver engagement, interactions, checklist items, application status, deposit status, source data, Ping, and vendor fields if integrated.

Then separate the behavior from the interpretation.

Define the threshold. Choose the allowed action. Assign the owner. Write the suppressions. Measure the outcome. Schedule a review date.

That sequence is slower than saying, "score everything." It produces work people can actually use.

For a more mature team, create a signal ledger.

Every active signal used in outreach, reporting, scoring, routing, or personalization should have:

1. Source.
2. Definition.
3. Refresh schedule.
4. Owner.
5. Allowed uses.
6. Suppressions.
7. Expiration or retirement condition.

That ledger becomes part of communications governance and adaptive enrollment management.

Without it, signal logic spreads across queries, rules, exports, portals, Deliver campaigns, source formats, and dashboards until nobody can see the whole system.

With it, the institution can govern what it built.

How To Study The Featured Sessions

Study the Holy Cross and ZeeMee session for signal structure. Watch how signals move from definition to activation across portals, campaigns, notifications, counselor queries, and leadership views.

Study the Roger Williams and Capture session for operational integration. Focus on apply, yield, and anti melt use cases, two way data movement, record visibility, query filters, and counselor workflow.

Study the Georgia session for experience design and measurement. The clearest signal strategy starts before the score, in the landing page, form, portal, tracking structure, and reporting path.

Study the yield volatility panel for strategic context. Watch how market shifts, direct admission, financial aid behavior, housing signals, portal personalization, and AI assisted analysis all increase the need for action ownership.

Read all four with one question in mind:

Where does the signal become an action?

That question will tell you whether the idea is ready to use, needs a pilot, or belongs in research for now.

Closing

The Summit shows institutions moving toward more sophisticated use of behavior, personalization, attribution, and yield intelligence. That direction is right.

The harder question is what teams are allowed to do with what they know.

That question is more useful than asking how much the institution can know.

The goal is a better enrollment operation. One that notices when a student needs help. One that reduces friction. One that respects student expectations. One that gives staff clear priorities. One that turns data into action without turning action into noise.

For every signal the team wants to use, finish the sentence:

When this happens, we believe it may mean this, so this owner may take this action through this channel within this timeframe, unless these suppressions apply, and we will judge success by this measure.

If the team cannot finish that sentence, the signal is not ready to drive outreach.

It may be interesting. It may belong in research. It may belong in a dashboard. It may belong in staff awareness. It is not ready to change student treatment.

Start with one signal, one decision, one owner, one action, and one measurement.

That is enough to begin.

Source Notes

1. S68, "Student Signals, Stranger Things, and the Path to Personalized Outreach," Moni Nann, Ben Parsons, Courtney Buckner, and Katy Remich, College of the Holy Cross and ZeeMee. Key source pages include 6 to 13 for signal foundation and student expectations, 16 to 20 for the signal

framework, 22 to 23 for define, structure, and activate, 24 to 28 for portal, email, scoring, and notification examples, and 29 to 33 for team use and implementation advice.
2. S36, "From Insight to Action: Using Behavioral Data to Power Targeted Campaigns," Aimee Celentano, Amanda Marsili, and Megan Gabbard, Roger Williams University Undergraduate Admissions and Capture Higher Ed. Key source pages include 8 to 18 for apply, yield, and anti melt use cases, 19 to 25 for two way data exchange and mappings, 26 to 27 for record tabs and queries, and 28 for lessons learned.
3. S05, "All the Small Clicks: Conversion Rate Optimization Lessons from Ads to Applications," Lillie Miller, Michael Doran, and Launa Smith, University of Georgia Undergraduate Admissions. Key source pages include 5 to 12 for conversion rate optimization, landing pages, calls to action, and forms, 21 to 30 for pixel setup and landing portal, 31 to 34 for the conversion rate optimization framework, and 36 to 39 for RFI, application, and email reporting.
4. S03, "Adapting to Yield Volatility: Using Data and Technology to Shape the Right Class," Catie Clark, Christie Smith, Olivia Nash, and Ally Metcalf, Niche, Alfred University, University of the Pacific, and The University of Tulsa. Key source pages include 5 to 10 for market shifts and strategy, 12 to 18 for Alfred direct admission measures, 20 to 29 for Pacific integrations, AI analysis, portal personalization, and melt strategy, and 31 to 33 for Tulsa direct admission growth.

Chapter 4 - Governed Personalization

There is a version of Slate personalization that looks strong for a few weeks.

The message is clever. The Liquid works. The audience seems reasonable. The content changes by major, student type, territory, event behavior, financial aid interest, parent context, and a special population someone remembered late in the process.

The message sends. People like it.

Three months later, nobody wants to edit it.

The logic is hard to follow. The population is unclear. The content block has branches inside branches. The query has exports that were needed once and then left behind. The subject line was approved for one audience and is now being reused for another. The suppression logic lives in one person's memory. The calendar in Slate has drifted away from the planning spreadsheet. The person who built the campaign says they can fix it, once they spend an hour remembering what they built.

That is weak personalization.

Governed personalization starts from a different standard. It treats communication as a system made of audiences, source definitions, content blocks, templates, Liquid, channels, suppressions, testing, reporting, ownership, and review.

The goal is relevance that can be understood, tested, reused, and maintained.

This chapter draws from five Slate Summit sessions:

1. "Mastering Flexible Drip Campaigns for Email and Print," presented by Mike Howie and Ross Struss from Hastings College.
2. "Solo Captain, Big Impact: Building Multi Phase Campaigns," presented by Joshua Frankenfield from Shippensburg University of Pennsylvania.
3. "The Hitchhiker's Guide to Content Blocks and Liquid Markup," presented by Christopher Kwan and Karin Ente from Technolutions.
4. "So You Want To Be A Rock and Roll Star?" presented by Matthew Schieren from the University of Chicago.
5. "Slate Powered Giving Tuesday: Multi Channel Personalization for Annual Giving," presented by Danielle Hupp and Daniel Porter from Kent State University.

The sessions come from different contexts. Some are admissions focused. One is advancement focused. Some are strategic. Some are technical. Together, they point toward the same lesson:

Personalization scales when institutions build governed, reusable decisions.

Extra branches can create extra risk. Strong personalization usually comes from clearer constraints, cleaner data, reusable content, and better review.

That is the central question for this chapter:

How can Slate teams make communication more relevant without creating a system that becomes too tangled to trust?

What Governed Personalization Means

Governed personalization is the practice of using student or constituent data to adapt messages, channels, timing, or experiences through documented, reusable, testable rules.

Four words matter in that definition.

Documented means another person can understand why the campaign exists, who receives it, what data changes it, and what outcome it supports.

Reusable means the work creates an asset rather than a private stunt. A template, content block, population, query, timing method, or scoring pattern can be used again with clear boundaries.

Testable means the team can review the main record types, blank values, edge cases, suppressions, links, accessibility, and outcomes before the campaign becomes student facing.

Rules means the personalization is tied to an approved decision about data use. The team knows which inputs may change the message and which should remain in reporting or staff awareness.

This moves personalization out of the narrow category of copywriting. Copy still matters. Tone matters. Clarity matters. The message a student reads is the visible surface of a larger operating system.

Underneath that message are populations, query bases, source codes, imports, portals, event records, material status, interaction records, content blocks, Liquid syntax, templates, print workflows, SMS rules, counselor tasks, chatbot source material, UTMs, reporting tags, approval steps, and review dates.

If that system is strong, personalization can feel useful and coherent. If that system is weak, personalization becomes fragile.

The mistake is treating personalization as freedom from structure. In practice, strong personalization needs a stable frame. A content block library, naming convention, campaign map, QA checklist, suppression policy, and owner list are creative infrastructure. They let the team write more relevant communication without forcing each new idea to become a new pile of exceptions.

Campaign Maturity Is A Build

The Hastings College session is useful because it treats campaign maturity as a steady build. Mike Howie and Ross Struss describe a small team moving from the idea of automation into a multi channel nurture system with more than forty automated touchpoints across three high school classes.

The pieces are practical: marketing populations, configurable join rules, timestamp filters, Deliver, Slate Print, one tap surveys, templates, and a restricted source chatbot.

The strongest lesson is the use of population timestamp filters. That detail can sound narrow, since it lives inside campaign mechanics. The concept is larger. Campaign timing has to survive real operating conditions.

Fixed day logic can be brittle. If a student enters a population late, leaves and returns, misses a window, or hits an unexpected process delay, a campaign tied too rigidly to day counts may behave in strange ways. Timestamp logic can make timing more durable because it looks at what happened and when.

That is governed personalization. The message may look simple. The timing rule is where the system becomes more dependable.

The Hastings session also connects email and print. That matters because many offices still treat print and digital communication as separate tracks. When the same campaign logic coordinates both, the team can design a message family. The student experience becomes less random. The institution can decide what each channel is for, when it should appear, and how it supports the next step.

The one tap survey pattern also matters. A small interaction can convert light interest into structured data. The response can inform a future message, portal element, counselor view, or campaign path. The value comes from turning a simple action into usable data.

The pattern is worth copying at the level of method rather than exact cadence:

1. Define the population.
2. Set timing rules that tolerate operational messiness.
3. Coordinate channels through the same campaign logic.
4. Use simple interactions to gather clearer interest data.
5. Review the campaign every cycle.

That is a small team version of communications infrastructure.

Start Where The Action Is Clear

Joshua Frankenfield's Shippensburg session speaks to another common reality. Many Slate communication systems depend heavily on one person.

That person may be the campaign architect, query writer, template editor, Liquid troubleshooter, QA lead, and emergency repair shop. This arrangement can work for a while. It creates a fragile office over time.

The session offers a practical path from one time emails into multi phase campaigns. It emphasizes transactional and application completion messaging, populations and rules, timestamp methods, query templates, MJML, Liquid, and bounded AI use.

One of its strongest lessons is where to start.

Begin with messages that have clear operational value.

Application completion messages are a strong starting point because the audience is defined, the purpose is clear, and the outcome can be measured. The same applies to missing materials, event follow up, admitted student next steps, transfer checklist completion, parent onboarding, and post deposit tasks.

These messages help students do something specific. That makes them easier to govern than broad nurture communication.

The audience is easier to explain. The data inputs are easier to inspect. The timing is easier to justify. The call to action is easier to test. The measure is closer to the student behavior the office actually wants.

That creates a good rule for teams moving into governed personalization:

Start where the action is clear.

Begin with a message family where everyone can answer:

1. Who receives this?
2. Why do they receive it?
3. What should they do next?
4. What data can change the message?
5. What should suppress it?
6. How will we know whether it helped?

Shippensburg's use of MJML also points to a broader design production issue. Every office needs a maintainable way to create responsive email. The exact tool can vary. The operating problem stays the same. If every message variation depends on one person hand editing code from scratch, the office has a succession risk. If templates are reusable and understandable, the office has an asset.

The same principle applies to AI. The useful role for AI in communication is bounded assistance: draft variants from approved material, improve clarity, suggest structure, help troubleshoot code, summarize documentation, or generate options for human review.

Keep AI away from policy invention, unsupported institutional claims, sensitive motivation inference, and direct student sends without review.

AI can speed up governed work. Governance still belongs to people.

Queries Shape Data. Liquid Renders It.

The Technolutions session on Content Blocks and Liquid gives the craft layer beneath governed personalization.

One principle deserves to become standard practice:

Queries should shape the data. Liquid should render the message.

That division of labor protects the system.

If too much presentation logic moves into queries, the query becomes harder to reuse. If too much data shaping moves into Liquid, the message becomes harder to test. The cleaner pattern is to make the query return well named, well structured fields, then use Liquid and content blocks to display the correct pieces.

This is governance disguised as syntax.

Clean export names are governance. Fallback copy is governance. Test records are governance. Closing tags is governance. A content block library is governance. A decision about when to use one dynamic message rather than several simpler messages is governance.

The common false victory is reducing twenty messages to one and then creating one message that nobody can safely edit. The team can say it simplified the campaign count. The actual system may be more complex.

The goal is appropriate abstraction.

Use content blocks when the structure is shared and specific pieces change. Use separate messages when the purpose, timing, audience, owner, tone, or risk is meaningfully different.

A dynamic message is useful when all major combinations can be tested and the logic remains readable. Several simpler messages are useful when they make ownership and QA clearer.

This is where the communication team needs technical judgment rather than a universal rule. Fewer assets can reduce maintenance. Fewer assets can also hide complexity inside one risky object. More assets can create clutter. More assets can also keep logic safer and clearer.

The right structure is the one the team can explain, test, and maintain.

Source Quality Sets The Ceiling

Matthew Schieren's University of Chicago session adds the wider marketing frame. Slate is a CRM. Weak source data, vague portal purpose, inconsistent channel use, and fuzzy reporting definitions stay weak until teams govern them.

This matters because many teams want to personalize before they have decided whether the data deserves that level of influence.

Source quality should shape allowed use.

High confidence student submitted interest data may support tailored academic content. A light purchased list may support an invitation to self identify. A completed RFI can support a more specific next step. A vague referrer may support reporting rather than direct personalization. Parent behavior may support parent communication. It should be handled carefully before being treated as student intent.

The source determines the confidence. The confidence determines the allowed action. The allowed action determines the message.

This is where communication governance and signal governance meet.

The University of Chicago session also highlights portal purpose, channel discipline, and reporting through Sources, Origin Sources, UTM parameters, referrers, and return analysis.

Those pieces can feel separate from copy. They are part of communication strategy.

If source definitions are inconsistent, campaign performance gets harder to evaluate. If UTM practices drift, attribution becomes guesswork. If a portal exists without a clear student purpose, it becomes extra maintenance. If SMS is used because it feels urgent rather than because the content deserves that channel, the channel loses trust.

Before a team personalizes, it should ask whether the data is strong enough, recent enough, specific enough, and appropriate enough to drive that student experience.

Personalization Across Channels

The Kent State Giving Tuesday session gives one of the strongest examples of coordinated personalization, even though the context is advancement.

Kent State had fifty eight possible funds and wanted to recommend the three most relevant funds for each constituent. The build used campaign data, affinity signals, prior giving, engagement, academic connections, weighted scoring, content blocks, Liquid, a custom portal with URL parameters, and caller talking points.

The pattern matters because the recommendation coordinated multiple channels.

The email pointed one way. The portal reinforced it. The caller saw the same context. The ranked recommendations created a shared experience rather than disconnected channel tactics.

Admissions and enrollment teams can adapt the pattern.

A student's academic interest can shape the email, the portal, the counselor context, the event invitation, and the parent message. A financial aid signal can shape the student resource path, the counselor note, and the parent support content. An event attendance pattern can shape follow up, portal content, and the next invitation.

The lesson is coordination.

Personalization becomes stronger when the same governed decision supports the communication environment. The strongest experience gives each channel a coherent role.

The Communications Governance Map

The practical tool for this chapter is a Communications Governance Map.

Use it for every major campaign family. It can be a document, spreadsheet, database entry, or Slate connected planning artifact. The format matters less than the discipline.

For each campaign, answer these questions:

1. Who is eligible?
2. Why are they included?
3. What decision or student action does this campaign support?
4. Which channels are involved?
5. Which fields or behaviors may change the content?
6. Which data sources are trusted enough for personalization?
7. Which templates, content blocks, snippets, forms, portals, print pieces, SMS language, or caller scripts are reused?
8. Who should be suppressed?
9. What happens when data is blank, stale, or conflicting?
10. Which test records need review?
11. What outcome matters?
12. Who owns maintenance?
13. When will the campaign be reviewed?

That map prevents common failures.

It stops teams from personalizing with weak data. It exposes missing fallback copy. It makes suppressions visible. It helps prevent three offices from messaging the same student about related topics on the same day. It forces edge case testing. It gives future staff a path back into the campaign logic.

The map also makes personalization explainable.

If a student receives a tailored message, the institution should be able to explain why the message made sense. The explanation should sound normal. If the internal explanation sounds strange, the student facing experience is probably too aggressive, too specific, or too poorly grounded.

The best test is plain:

Could a staff member explain this outreach to the student in one sentence?

"You told us you were interested in engineering, so we sent related academic resources" is easy to explain.

"You clicked scholarship content three times after 10 p.m., so we changed your counselor priority" is a much harder case.

The issue is not whether Slate can see the behavior. The issue is whether the institution should use the behavior that way.

Three Design Tensions

Governed personalization requires judgment. Three tensions show up often.

One Campaign Or Many

Teams often ask whether one dynamic Deliver message is better than several simpler messages.

There is no universal answer.

One dynamic message is usually better when the audience shares the same purpose, timing, owner, and structure, and only a few sections vary. Separate messages are usually better when the purpose, timing, owner, tone, risk, or QA path changes.

The guiding standard is maintainability. Choose the clearest structure the team can safely test.

Automation Or Human Review

Automation is necessary for scale. Hastings and Shippensburg both show small teams using automation thoughtfully.

The level of review should match the sensitivity of the message.

Application completion reminders, event reminders, and basic interest nurture can often run through strong automated rules. Financial aid concern messages, parent messages based on student behavior, high priority counselor interventions, and AI supported campaigns usually need stronger review.

The more consequential the message, the clearer the review process should be.

Creativity Or Consistency

Fresh content matters. Students can tell when a message was copied forward without care.

Consistency matters too. If every message is invented from scratch, tone drifts, claims change, deadlines get phrased differently, accessibility is missed, and reporting tags scatter.

Governance gives creativity a frame. It keeps the work usable.

A content block library, naming convention, QA checklist, campaign map, and source policy make the craft repeatable.

AI In Communications Governance

AI belongs inside the same discipline as every other communication tool.

It can help draft, compare, edit, explain, summarize, classify, and troubleshoot. It still needs approved source material. It still needs human review. It still needs clear boundaries around student data, policy language, and claims.

For communications work, AI use can be grouped into three categories.

Low risk uses include rewriting approved content for clarity, producing draft variants from verified material, suggesting subject lines, explaining Liquid syntax, troubleshooting HTML, or summarizing campaign documentation.

Review required uses include generating new campaign copy, creating audience specific claims, modifying Liquid or HTML, producing chatbot source content, or recommending segmentation logic.

Formal approval uses include financial aid language, policy statements, inferred student motivations, sensitive population logic, unsupported outcome claims, chatbot deployment, and anything that would send directly to students.

This standard is pro quality. It gives the team a way to use AI without letting it set institutional policy by accident.

Measurement Beyond Opens

A governed campaign should be measured against its purpose.

Opens and clicks are diagnostic. They are rarely the final goal.

For application completion, measure completion. For event campaigns, measure registration and attendance. For yield campaigns, measure meaningful next steps, deposit, aid appointments, housing movement, portal checklist progress, or counselor interaction. For source campaigns, measure inquiry quality, application rate, admit rate, deposit rate, and cost per outcome where possible. For print, measure whether the combined channel strategy improves downstream behavior. For one tap surveys, measure the response and what happens after the response. For chatbots, measure answer accuracy, escalation, usage, and staff impact.

The point is simple: the measure should match the action.

A campaign with clean governance and weak outcomes needs revision. A campaign with strong outcomes and messy logic may need governance before it can safely scale.

Both performance and maintainability matter.

A Staff Exercise

The best way to teach governed personalization is to build the map before writing the copy.

Give the team one campaign scenario:

1. An application completion campaign.
2. An admitted student yield campaign.
3. An event follow up series.
4. A parent support sequence.
5. A post deposit onboarding campaign.

Then ask the team to complete the Communications Governance Map.

Who is the audience? What is the purpose? Which data is trusted enough to personalize? What content changes? What stays fixed? Who is suppressed? What test records are needed? What outcome matters? Who owns it after launch?

Then give the team sample records:

1. A student with strong academic interest.
2. A student with financial aid engagement.
3. A student with parent only engagement.
4. A student with high portal activity.
5. A deposited student missing onboarding items.

Ask what changes for each record.

Does the content change, or only the channel? Does the owner change? Does the student get suppressed? Is a portal change better than another email? Does the signal justify action, or only awareness?

That exercise teaches personalization as decision design.

It also exposes weak definitions. If nobody knows what financial aid engagement means, hold personalization until the definition is clear. If nobody knows whether portal activity came from a student or parent, be careful. If source reliability is unclear, use the data for reporting first. If nobody owns parent communication, define ownership before building the campaign.

The training exercise becomes a governance audit.

How To Study The Featured Sessions

Study the Hastings session for campaign maturity, flexible timing, and the coordination of email and print. Focus on how a small team turned campaign logic into a reusable operating pattern.

Study the Shippensburg session for the solo builder problem. Notice the emphasis on starting with operationally clear messages, using templates, creating a maintainable production workflow, and placing AI inside review.

Study the Technolutions Content Blocks and Liquid session for craft discipline. Pay attention to the division between data shaping and presentation, reusable content, test records, loops, fallback copy, and debugging.

Study the University of Chicago session for source quality, channel discipline, portal purpose, and attribution. This is the foundation that determines whether personalization has enough truth underneath it.

Study the Kent State session for multi channel coordination. The fund recommendation model is less important than the operating pattern: one governed recommendation can support email, portal, and human outreach at the same time.

Read the five sessions with one question in mind:

Can the personalization be explained, tested, reused, and reviewed?

If the answer is yes, the campaign is becoming a tool. If the answer is no, the campaign may still send, though it is creating future debt.

Closing

Personalization is often described as a way to make communication feel more human. That can be true.

Poorly governed personalization does the opposite. It exposes the machinery. It references details students fail to recognize. It repeats itself. It contradicts other messages. It makes the institution look less attentive.

Governed personalization uses data with restraint. It reuses approved content. It respects source confidence. It makes timing visible. It protects students from noise. It gives staff a shared operating model. It can be tested. It can be explained. It can survive the person who built it.

That is the standard.

Pick one important campaign and write its governance map.

A campaign remains unfinished until the team can define the audience, purpose, personalization inputs, suppressions, QA plan, owner, review date, and success measure.

It may still send. It may look good. It may even perform well for now.

The stronger move is to turn it into a tool.

Source Notes

1. "Mastering Flexible Drip Campaigns for Email and Print," Mike Howie and Ross Struss, Hastings College. Key source pages include 3 to 5 for campaign maturity, 6 to 10 for populations, rules, and campaign planning, 12 to 16 for timestamp scheduling and print, 17 for template tool choice, 18 to 21 for one tap survey design, and 22 for the restricted AI chatbot.
2. "The Hitchhiker's Guide to Content Blocks and Liquid Markup," Christopher Kwan and Karin Ente, Technolutions. Key source pages include 3 to 16 for content block types and examples, 17 to 45 for Liquid objects, tags, filters, and loops, and 46 to 59 for lifecycle, multi interest examples, debugging, and practice guidance.
3. "Solo Captain, Big Impact: Building Multi Phase Campaigns," Joshua Frankenfield, Shippensburg University of Pennsylvania. Key source pages include 6 to 9 for campaign stages and maturity, 13 to 20 for populations, rules, timestamp logic, and query templates, 23 to 37 for MJML, templates, Liquid, and query library, 39 to 40 for sample cadence, and 41 to 43 for AI guidance.
4. "So You Want To Be A Rock and Roll Star?" Matthew Schieren, University of Chicago. Key source pages include 8 for the five theme frame, 11 to 22 for source quality and data handling, 24 to 29 for campaigns and channel choice, 31 to 37 for portal decision questions, and 40 to 47 for source, Origin Source, UTM, referrer, and return analysis.
5. "Slate Powered Giving Tuesday: Multi Channel Personalization for Annual Giving," Danielle Hupp and Daniel Porter, Kent State University, Philanthropy and Alumni Engagement. Key source pages include 14 for the four step build, 15 to 18 for fund map and scoring, 19 to 22 for person results, email, portal, and caller view, 24 for crowdfunding datasets, and 25 for campaign total.

Chapter 5 - Events and Portals as Signal Systems

An event ends too early when the last artifact is an attendance count.

The open house happened. The admitted student day happened. The virtual session happened. The campus tour happened. The student showed up or missed it. The attendance list may have been exported. A thank you message may have sent. A report may have recorded the number.

That is a thin version of event work.

Modern Slate practice points toward a stronger view. An event is a designed experience that can help a student move forward and help the institution understand what kind of next action is warranted. A portal works the same way. It can guide a student through a process, organize resources, capture intent, reduce confusion, and show where the next step is breaking down.

Events and portals are signal systems.

They can show intent, friction, topic interest, family involvement, schedule choice, uncertainty, readiness, and follow through. They can help staff understand who registered, who attended, what they chose, what they missed, what they returned to, and what should happen next.

The craft is choosing which evidence matters and which action the evidence should inform.

This chapter draws from five Slate Summit sessions:

1. "Technolutions Training: Building a Keystone Event," presented by Robin Lancaster and Ben Willner from Technolutions.
2. "Where's My Camper: A Real Time Attendance Tracking Solution," presented by Caleb Whitman and Ryan Herman from Southern Adventist University Undergraduate Admissions and Southern Adventist University Graduate Admissions.
3. "Event Schedules II: Personalized Schedules Without Custom SQL," presented by Nathan Vick from Asbury University.
4. "We Walk the Line: Cutting Costs and Complexity in Event Management," presented by Taylor Frechette, Michael Doran, and Kelly Coffman from the University of Georgia Undergraduate Admissions.
5. "FanTABstic Portals: How to Build Them and Ways to Use Them," presented by Sarah Bundschuh and Rachel Haywood from the University of Delaware Undergraduate Admissions.

The sessions cover the full arc of event and portal work: event foundations, live operations, schedule personalization, cost discipline, and portal experience design.

Together, they point toward one lesson:

Events and portals are becoming measurement systems. They help institutions see where students are leaning in, where they are getting stuck, and where the next action should be more specific.

The practical question is simple:

How can Slate teams capture useful event and portal signals while keeping the system light enough to operate?

Event Telemetry

The traditional event question is familiar:

How many people registered, and how many attended?

That still matters. Registration and attendance are foundational. If the office stops there, it misses much of the usable evidence.

The better question is:

What did this event tell us about the student's next step?

A registration tells the office something. Attendance tells the office more. A no show tells the office something different. A student who chooses a financial aid session over a student life session tells the office something. A student who attends the academic breakout and skips the optional tour tells the office something. A parent who engages with cost content may tell the office something. A student who logs into the portal after the event may tell the office something. A student who attends several events and leaves the next required step unfinished may be telling the office something else.

Those behaviors need cautious reading. Behavior gives evidence. It leaves room for uncertainty. The value comes from designing an event and portal system that gives staff better evidence than a flat attendee list.

That is event telemetry.

Event telemetry is the structured evidence an institution collects before, during, and after an event so it can understand participation, friction, interest, and action needs.

Before the event, telemetry may include invitation source, registration timing, guest count, chosen sessions, student type, academic interest, submitted questions, portal return, and changes to the registration record.

During the event, telemetry may include check in, attendance, walk up registration, session participation, staff notes, group movement, schedule changes, capacity pressure, and service issues.

After the event, telemetry may include follow up clicks, portal activity, application movement, deposit movement, checklist completion, additional event registration, counselor interaction, survey response, and parent engagement.

That gives the team a full loop:

1. Pre event intent.
2. In event behavior.
3. Post event action.

The strongest event systems connect those three phases. They do this with restraint. They capture the signal the team is ready to use.

Keystone Events Need Foundations

Robin Lancaster and Ben Willner's session on building a keystone event is useful because it treats major events as load bearing systems.

A keystone event carries real institutional weight. It may be a major admitted student program, a large open house, a signature visit day, a program specific event, or a large recruitment experience with several audiences moving through the day. The build needs more care because the event affects student experience, staff coordination, communications, reporting, and later follow up.

The foundation determines how useful the later signals can be.

If registration categories are clear, the team can interpret the audience. If guests are handled intentionally, parent and supporter behavior becomes easier to understand. If attendance is captured reliably, the office can separate actual participation from light intent. If schedule options are structured well, session choice can support better follow up. If communication is coordinated, the student experience feels coherent. If event records connect cleanly to later reporting, the event becomes part of the student record in a meaningful way.

Many institutions want insight after an event. They want to know which parts of the day influenced conversion, which academic sessions mattered, which no shows came back later, which guests signaled family concern, and which follow up paths worked.

Those answers are hard to recover after the build ignores the evidence.

If schedule choice was stored poorly, the team has little usable context. If attendance was incomplete, no show recovery becomes unreliable. If guest handling was inconsistent, family participation is hard to interpret. If the event record has no clear relationship to portal behavior, staff lose the chance to see what happened after the student left.

The stronger approach is to design the event with the next action in mind.

For a major event, planning should include this question:

What should the event teach us once students begin participating?

That question affects the registration form, schedule options, event populations, check in plan, staff roles, follow up queries, portal updates, and reporting.

It also keeps the team from collecting data because it is available. More data creates more maintenance. The goal is usable evidence.

The first principle is clear:

Capture the signal you are prepared to use.

Live Signals Need Live Response

The Southern Adventist session on real time attendance tracking brings event signals into live operations.

Some event data becomes useful later. Real time attendance data is different because it can change what staff do while the event is still happening.

If staff can see that a participant has not checked in, someone may be able to call, text, redirect, or help. If a group is missing from a session, the team may be able to locate them before the next transition. If an event serves minors, families, or structured groups, knowing where people are has service and safety value beyond later reporting.

That is the force of the Southern Adventist example.

Real time tracking has value only when the team has a real time response plan. Otherwise the event produces a dashboard that people admire after the moment has passed.

The useful pattern is:

Track the things staff can act on while action is still possible.

That requires role clarity. Someone watches the data. Someone responds. Someone contacts the student or group leader. Someone updates the record. Someone handles exceptions. Someone decides when a missing check in needs response.

Many events need simple attendance and a clean follow up path. A camp, group visit, or large admitted student program may need live awareness.

The second principle is clear:

If the signal is live, the response has to be live too.

Personalized Schedules As Experience And Evidence

Nathan Vick's session on personalized schedules without custom SQL sits at a useful intersection: student experience, maintainability, and data value.

A personalized schedule can make an event feel more relevant. A student can see sessions that match academic interest, student type, visit goals, family needs, or registration choices. Staff can shape the day with clearer paths. The event gives the student a clearer route than a generic agenda would provide.

The title matters because "without custom SQL" speaks to maintainability. Event systems become fragile when schedule logic depends on work only one or two people can maintain. When schedule personalization can be built through accessible Slate tools and clear configuration, the institution gets relevance with less technical fragility.

The lesson is fit.

If every student's day is mostly the same, schedule personalization may add effort without improving the experience. If the event has meaningful tracks, competing sessions, student type differences, academic breakouts, family options, or required and optional components, personalized schedules can make the day clearer.

The schedule itself also becomes a signal.

A student who chooses nursing, financial aid, and a parent panel is showing a different pattern from a student who chooses engineering, honors, and undergraduate research. That pattern should still be read with caution. Schedule choice gives one piece of context. It can support a better next step.

A strong event follow up should know more than whether the student attended. It should know enough about the experience to continue the conversation.

That may mean an academic resource tied to a chosen session. It may mean a parent support message after a family focused path. It may mean a counselor note when a student attends a high intent session and later returns to the portal. It may mean no additional segmentation because the event did not generate meaningful differences.

The third principle is clear:

Personalization is useful when it improves both the student experience and the evidence for follow up.

Complexity Has To Earn Its Place

The University of Georgia session on cutting costs and complexity belongs in the same conversation as real time tracking and personalized schedules.

Event teams can build impressive machinery in Slate. Every added layer has a cost. More registration options mean more logic. More schedule paths mean more testing. More

communication branches mean more copy. More data capture means more definitions. More staff roles mean more training. More moving parts mean more failure points.

Complexity is worth spending when it returns real value.

For a major admitted student program, personalized schedules, reliable attendance tracking, segmented follow up, and a portal hub may be worth the effort. For a small virtual information session, that same machinery may be too much. For a high volume campus tour program, simplicity and repeatability may matter more than personalization. For an event with minors or structured groups, live tracking may carry operational value beyond enrollment analytics.

The Georgia session reminds teams that event strategy is also cost strategy. Time, staff attention, training, testing, content maintenance, and reporting all have a cost.

A simple complexity test helps:

1. Does this feature improve the student experience?
2. Does it reduce staff burden?
3. Does it capture a signal the team will use?
4. Does it improve a decision the institution actually makes?
5. Does it create a maintenance burden that the team can own?

If a proposed feature fails to answer at least one of the first four questions clearly, it is probably a clever extra with weak system value. If it fails the fifth question, the team is borrowing trouble from the next cycle.

The fourth principle is clear:

Complexity has to earn its place.

Portals Extend The Signal Loop

Sarah Bundschuh and Rachel Haywood's session on portals belongs with the event sessions because portals can become the connective tissue before, during, and after events.

A portal can organize information. It can guide next steps. It can reduce confusion. It can display personalized content. It can collect intent. It can make event information easier to access. It can capture return behavior. It can give students a place to continue after the event ends.

That makes portals experience infrastructure.

Tabs matter because they give structure. A student can move through academics, events, next steps, resources, community, financial aid, or enrollment tasks without facing one long wall of content. The tab structure creates a map.

A portal still needs a purpose. It should answer a real experience problem.

Maybe event registrants need a day of hub. Maybe admitted students need deadlines, checklist items, event options, and resources in one place. Maybe parents and supporters need clearer guidance. Maybe a program has too many disconnected links. Maybe staff need one consistent place to send students after a call.

The portal becomes useful when it creates coherence.

Once students use it, the portal also produces evidence. Which tabs do students return to? Which calls to action do they use? Which resources attract attention? Which steps remain incomplete? Which sections create questions? Which links support conversion? Which portal experiences reduce repetitive staff questions?

The point is practical learning. The institution should learn where the experience helps and where it breaks down.

The fifth principle is clear:

Portals should guide the student and teach the institution.

The Event Signal Loop

Put the five sessions together and a mature model appears.

A keystone event gives the foundation. Real time attendance tracking gives live responsiveness. Personalized schedules give relevance and stronger context. Cost and complexity discipline keeps the system from overbuilding itself. Portals organize the larger experience and extend the loop beyond the event.

The event begins when the student first receives or finds the invitation.

The portal begins when the institution decides what experience problem it is supposed to solve.

The event should end only after the next appropriate action has happened.

That action might be a same day message, academic interest follow up, parent resource, financial aid invitation, counselor note, no show recovery, portal update, staff task, or clean reporting record.

The key is that the action should match the signal.

An attendee and a no show should receive different follow up. A student who attended an academic breakout may need a different next step from someone who only attended a general session. A student who registered with a parent may need different support than one who came alone. A student who attends repeatedly and stalls may need different attention from one who has no recent engagement.

This requires judgment about meaningful differences.

Most teams can start with three event groups:

1. Attendees.
2. No shows.
3. High intent participants.

Attendees should receive follow up that acknowledges the event and points to the next logical step. No shows should receive a respectful recovery path. High intent participants should move into a more specific next action when their behavior supports it.

That simple model is strong enough to begin.

From there, a team can add nuance where it is justified. Schedule choice may matter. Academic interest may matter. Distance traveled may matter. Parent attendance may matter. Portal behavior after the event may matter.

Start with the loop first:

1. Before the event, what intent did the student show?
2. During the event, what participation did the student demonstrate?
3. After the event, what should happen differently because of what the team learned?

That is the Event Signal Loop.

Applying The Loop To One Event

The practical resource for this chapter is a one event signal worksheet.

It can live in a document, spreadsheet, project brief, or Slate adjacent planning artifact. The format matters less than the habit.

For one event, answer these questions:

1. What is the event supposed to help the student do?
2. What is the event supposed to help the institution learn?
3. Which signals will be captured before the event?
4. Which signals will be captured during the event?
5. Which signals will be reviewed after the event?
6. Which signals will be left unused because the team still lacks an action plan?
7. What happens differently for attendees?
8. What happens differently for no shows?
9. What happens differently for high intent participants?
10. Which portal pages or tabs support the event?

11. Which staff roles are involved during the event?
12. Which follow up actions have owners?
13. Which outcome will be reviewed later?
14. What should be simplified before the event runs again?

The sixth question matters. A signal that the team captures and never uses creates clutter. A signal that staff struggle to explain creates mistrust. A signal that generates action without review creates risk.

Selective tracking is usually better than maximum tracking.

Capture what you will use. Use what you can explain. Explain what staff can maintain. Maintain what improves the student experience or a real institutional decision.

A Practical Example

Imagine a large admitted student event.

The registration form captures academic interest, guest count, and optional breakout selections. The student receives a confirmation message and a portal tab with schedule details. On the day of the event, check in is captured. Session attendance is captured for the most meaningful breakouts, not every minor movement. Staff can see enough live data to solve actual operating problems.

After the event, attendees receive follow up tied to their chosen track. No shows receive a recovery message with alternate ways to engage. Students who attended a high intent session and returned to the portal afterward appear on a counselor review list. The event report does more than count attendance. It shows which signals led to which actions and which outcomes moved later.

That is an event signal system.

Now imagine a weaker version.

The team builds a polished event. Registration works. The room is full. The day feels successful. Attendance is incomplete. Session choices are disconnected from follow up. No shows receive the wrong message. The portal is stale. Counselors lack a clear view of what students did. The year end report shows only registration and attendance counts.

The event may still have value. The institution learned less than it could have.

The gap between those examples is design.

The stronger team decided ahead of time what evidence mattered and what action the evidence should trigger.

What Teams Should Do Next

Start with one event.

Choose an event that matters enough to justify the exercise. Avoid the full event portfolio at first.

Write down what the event is supposed to teach the institution. That phrase may feel strange at first, which is why it is useful. The event should teach the student something. It should also teach the office something about the student's needs, interests, concerns, or readiness.

Then identify three or four signals worth capturing. Registration source, attendance, session choice, guest participation, portal return, or post event click may be enough. The right signals depend on the event.

Next, decide what should happen differently for attendees, no shows, and high intent participants. If follow up is identical for everyone, the event data is staying decorative.

Then assign owners. Someone owns attendee follow up. Someone owns no show recovery. Someone owns counselor review. Someone owns the portal update. Someone owns the report.

Then rate the complexity. A simple event can have simple capture and simple follow up. A high stakes event can justify more. The rating is about fit.

Finally, review the event after follow up has had time to work. Did the signals help? Did staff understand their roles? Did students use the portal? Did no show recovery produce another action? Did high intent participants move forward? What should be simplified before the next cycle?

Event systems improve through cycles. The goal is a loop that can be inspected and improved.

The Standard

Events and portals are where students reveal questions, preferences, hesitation, enthusiasm, and friction.

A well designed event gives students a clearer sense of the institution.

A well designed portal gives students a clearer path through the process.

A well designed signal system gives staff a clearer reason to act.

Those three things belong together.

The standard is simple:

An event should produce a useful next action, or a clearer reason that no action is needed.

A portal should help the student move, or help the institution understand where movement has stalled.

Attendance reporting is a start. Event strategy goes further. It connects intent, participation, portal behavior, follow up, ownership, and review.

For the next event, begin with three questions:

1. What did the student tell us before the event?
2. What did the student show us during the event?
3. What should happen after the event because of what we learned?

That is the Event Signal Loop.

Source Notes

1. "Technolutions Training: Building a Keystone Event," Robin Lancaster and Ben Willner, Technolutions. This session frames major events as structured builds where registration, capacity, communication, check in, schedule design, guest handling, reminders, and follow up need to work together.
2. "Where's My Camper: A Real Time Attendance Tracking Solution," Caleb Whitman and Ryan Herman, Southern Adventist University Undergraduate Admissions and Southern Adventist University Graduate Admissions. This session shows how live attendance data can support action during an event, especially when staff roles and response paths are clear.
3. "Event Schedules II: Personalized Schedules Without Custom SQL," Nathan Vick, Asbury University. This session shows how personalized schedules can improve relevance while keeping the build maintainable through accessible Slate configuration.
4. "We Walk the Line: Cutting Costs and Complexity in Event Management," Taylor Frechette, Michael Doran, and Kelly Coffman, University of Georgia Undergraduate Admissions. This session keeps event sophistication tied to staff capacity, cost, repeatability, and return on effort.
5. "FanTABstic Portals: How to Build Them and Ways to Use Them," Sarah Bundschuh and Rachel Haywood, University of Delaware Undergraduate Admissions. This session shows how portals can organize resources, guide action, and extend the student experience beyond a single event moment.

Chapter 6 - The Hidden Operating System

The riskiest object in a Slate instance is often the one everyone forgot exists.

The new rule usually receives careful testing. The workflow that just went through a rebuild has attention. The new assignment list has been discussed for two weeks. The bigger risk often sits in the older layer: a rule with a vague name, a workflow trigger created during a deadline crunch, an assignment list that still routes records through old territory logic, a permission group that made sense when the staff was smaller, or a dataset that feeds a report that feeds a decision nobody has mapped.

Every Slate instance has a visible layer. Forms, portals, events, Deliver messages, Reader bins, reports, dashboards, queries, and records are the objects people discuss because staff can see them. They appear in daily work. They have screens, links, tabs, folders, and names.

Beneath that layer sits another system. Rules, workflows, assignments, permissions, source logic, legacy decisions, inherited naming conventions, exceptions, patches, workarounds, and quiet automations shape the work every day. They decide who sees what, which records move, what gets assigned, when staff receive tasks, which messages are suppressed, which fields change, which doors open, and which doors stay closed.

That layer is the hidden operating system of a Slate instance.

Hidden logic is necessary. A mature Slate instance needs more than visible manual action. Rules, workflows, assignments, and permissions make work repeatable. They help offices scale. They protect consistency. They reduce repetitive handling. They make student records move through complex institutional processes.

The risk begins when the hidden layer becomes difficult to explain.

A team needs a clear explanation of the logic before it can safely improve a process. New work gets built on top of old assumptions. Staff develop workarounds around unclear statuses. Permission groups grow by convenience. Assignment lists become daily staff reality, even when their logic no longer matches the work. Reports inherit definitions nobody remembers setting.

This chapter draws from five Slate Summit sessions:

1. "Behind the Music: The Hidden Rules Powering Your Instance," presented by Nina Lifshey, Darian Quinn, and Drew Griffin from Carnegie and Rockhurst University.
2. "Assignment Lists & Cadences," presented by Traci Giovinazzo, Colby Vere, and Benjamin Szabo from Technolutions.
3. "Workflows: From Legacy to Universal," presented by Ken Higgins from Technolutions.
4. "Written in Slate (Not Stone): Excavating Legacy Decisions and Building for Scale," presented by Raissa Diamante and Erin Gore from Harvey Mudd College and Huron.
5. "Don't Forget to Lock the Doors: Enhancing Security and Data Protection," presented by Nathan Vick, Emma Ortiz, Amy Shaiman, and Matthew Haugen from Asbury University, Miami University of Ohio Advancement, and Baylor University Undergraduate Admissions.

These sessions cover different objects and concerns. Rules are different from workflows. Assignment lists are different from permissions. Legacy decisions are different from security settings. Inside a real institution, they behave like one connected system. Together, they determine how the office operates.

The central question is practical:

Can the team make hidden logic visible enough to own, review, teach, and improve?

The Layer Beneath The Screens

Slate teams often talk about what people can see. They discuss a portal page, a Deliver message, a Reader bin, a report, a form, or an event. Those objects matter because they are the visible surfaces of the system.

The visible object often depends on hidden logic.

A portal tab may depend on role permissions, population logic, checklist rules, material status, and source definitions. A Deliver campaign may depend on suppressions, Liquid exports, source confidence, interaction rules, and population membership. A Reader bin may depend on assignment logic, application status, territory ownership, reader permissions, and review workflow. A dashboard may depend on hidden joins, lookup values, legacy source handling, and scheduled rules.

This is why Slate maintenance can feel confusing. The problem a staff member sees on screen may have a cause several layers below the surface.

A student appears on the wrong list. The issue might be a stale territory table, a changed school relationship, an old population rule, a source value, a material status, a workflow transition, or a permission boundary. A message goes to the wrong audience. The issue might be a query, a missing suppression, a changed field, an old campaign assumption, or an export name reused in a new context.

The hidden operating system is the part of the instance that explains why visible work behaves the way it does.

Strong Slate teams learn to read that layer. They ask which rule acted, which workflow moved the record, which assignment list surfaced it, which permission allowed it, which source value drove it, and which old decision shaped it.

That kind of reading turns troubleshooting into stewardship.

Rules Carry Decisions Forward

"Behind the Music: The Hidden Rules Powering Your Instance" is a useful starting point because rules are often the backing track of a Slate instance. Staff may miss them directly. They feel the rhythm rules create.

A rule can make work faster. It can protect consistency. It can reduce repetitive updates. It can make a process more reliable. It can also preserve an old assumption long after the original reason has faded.

Rules are easy to create under pressure. A deadline appears. A special population needs handling. A field needs to be set. A new process needs a temporary patch. Someone solves the problem. The cycle moves on. The rule remains.

That pattern is normal operational life. The issue is review.

A temporary rule needs a future. It should become a standard, receive documentation, get rebuilt into a cleaner process, or retire when the need passes. When none of those things happen, a practical fix becomes hidden debt.

A strong rule inventory does more than list names. It explains the work.

For each important rule, the team should know:

1. What the rule does.
2. What triggers it.
3. Which population it affects.
4. Which fields, records, messages, tasks, or processes it changes.
5. Which process depends on it.
6. Who owns the business decision behind it.
7. Who understands the technical configuration.
8. When it was last reviewed.
9. What could break if it stopped running or ran incorrectly.
10. Whether it should be kept, rebuilt, consolidated, restricted, or retired.

The most useful part of that inventory is plain language. A rule name can tell a builder where to look. Plain language tells the office why the rule exists.

The rule record should answer a basic question:

What decision does this rule carry forward?

That question changes the tone of cleanup. The team is recovering institutional memory rather than hunting mistakes.

Legacy Decisions Need Context

"Written in Slate (Not Stone): Excavating Legacy Decisions and Building for Scale" gives the right posture for inherited Slate work. The word excavating matters because legacy decisions are usually buried context rather than obvious failure.

Many old objects were created for sensible reasons. A staff member solved a real problem with the tools, time, knowledge, and pressure available at the moment. A later team may inherit the object without inheriting the reason.

That is where risk grows.

An old rule may still be exactly right. An old workflow may still reflect the process. An old report may still support an important decision. An old permission model may still protect sensitive work. Age alone says little about object quality.

Silence creates the problem.

Objects that have run quietly for years can become exempt from review. Staff trust them because they exist. New builders avoid them because they are unclear. Leaders may assume the system is working because no one has named the risk.

A legacy decision log helps restore context.

For each inherited object or process, capture:

1. The object name.
2. The Slate area or object type.
3. The original purpose, if known.
4. The business decision behind it.
5. The population affected.
6. The current process it supports.
7. The owner today.
8. The related reports, messages, workflows, permissions, or assignments.
9. The risk if it stays unchanged.
10. The next action.

Some items should be kept. Some should be retired. Some should be rebuilt. Some should simply get better names and notes. The work is making the system honest about what it is doing.

Institutional memory belongs in the system and in the people who maintain it.

Assignment Lists Are Work Routing

"Assignment Lists & Cadences" belongs in this chapter because assignment structures shape daily staff behavior. An assignment list is a work routing decision.

It determines which records appear in front of which staff members. It affects counselor attention, workload balance, territory management, outreach expectations, and the rhythm of follow up. Over time, the list can become the work itself.

That gives assignment lists real power.

If a list is based on territory, the territory logic needs ownership. If it is based on stage, the stage definition needs clarity. If it is based on behavior, the behavior needs confidence. If it is based on priority, the priority definition needs governance. If it is paired with a cadence, the cadence should reflect the action the office actually wants staff to take.

Staff often see the records before they see the logic. If the logic is strong, the list helps staff focus. If the logic is stale, unclear, or too broad, staff time gets misdirected.

That makes assignment list documentation more than technical housekeeping.

For each important assignment list, the team should know:

1. What the list is for.
2. Who should use it.
3. How a record enters.
4. How a record exits.
5. How often the list refreshes.
6. Which staff action the list expects.
7. Which records should be excluded.
8. Which report or review checks whether the list is useful.
9. Who owns the logic.
10. When the logic will be reviewed.

Cadences raise the stakes because they move from surfacing records to structuring action. A cadence can create consistency and prevent missed outreach. It can also encode the wrong rhythm at scale.

The better question asks whether the cadence reflects the follow up rhythm the team wants to stand behind.

A cadence should encode judgment.

Workflows Are Process Agreements

"Workflows: From Legacy to Universal" points toward a broader modernization pattern. Workflows translate process into structure. They move records, support review, coordinate tasks, show status, and help staff understand where something stands.

Workflow modernization can become a technical conversion project. That misses the larger opportunity.

A workflow is a process agreement.

It says what starts the process, who acts, which status means waiting, which status means done, which status means stuck, which notes matter, which transitions are allowed, and which outcome the office expects.

When a workflow has existed for a long time, staff often adapt around it. They learn where to click. They learn which status names have drifted from their plain meaning. They learn which steps are skipped. They learn the workaround that makes the process move.

Familiarity can hide confusion.

A workflow review should begin with the lived process:

1. What is supposed to happen?
2. What actually happens?
3. Where does the record enter?
4. What triggers movement?
5. Who acts at each stage?
6. Where does the process wait?
7. What counts as completion?
8. Which steps still matter?
9. Which steps represent old process decisions?
10. What should staff see in Slate so the process feels clear?

That review should include Slate builders and process owners. The builder can explain what the configuration does. The process owner can explain what should happen in practice. The strongest workflow sits where those explanations match.

Before rebuilding hidden logic, recover the process behind it.

Permissions Are Governance

"Don't Forget to Lock the Doors" belongs in this chapter because permissions are one of the most important parts of the hidden operating system.

Permissions shape daily work constantly. They decide who can see sensitive data, export records, edit objects, impersonate, change configuration, access populations, run queries, view records, or make a small mistake with large consequences.

Permissions are governance decisions.

The hard part is that security often competes with convenience. A staff member needs access. A deadline is close. A team wants speed. Someone asks for a broad permission because a narrow one takes more time to design. Sometimes quick access is reasonable. Without review, it becomes permission drift.

Permission drift happens when access expands and rarely contracts. A staff member changes roles and keeps old access. A temporary permission becomes permanent. A group inherits access that no longer fits. A broad permission is copied because it is faster than designing a better group. Someone can still do something because nobody remembered to remove the access.

The answer is a review cadence.

A permission review cadence should include role changes, departures, new responsibilities, sensitive permission groups, export access, administrative access, and permissions connected to financial, identity, academic, or protected data.

The principle is plain:

Staff should have the access they need for the work they are expected to do.

Good security is good staff support. It reduces the chance that someone sees data they should avoid handling, changes something by accident, or becomes responsible for a risk they never meant to carry.

Exceptions will exist. Some work requires special access for a period of time. The exception needs an owner, reason, scope, and review date. Undocumented exceptions create the risk.

Access is part of the operating system. The doors matter.

The Process Trace

Rules, workflows, assignment lists, permissions, and legacy decisions may live in different areas of Slate. In practice, they interact.

A rule updates a field. That field changes an assignment list. The list creates a staff expectation. A workflow uses the same field as a trigger. A permission group determines who can see the workflow. A legacy report counts records based on the outcome. A staff member builds a new process from that report. Two years later, someone asks why a record moved the way it did, and the answer requires a trace across the instance.

That is the hidden operating system in motion.

A generic inventory has value. A process trace usually teaches more.

A generic inventory says, "Here are all the rules."

A process trace says, "Here is how this office actually runs."

Start with one process that matters:

1. Application review.

2. Event follow up.
3. Inquiry assignment.
4. Staff task routing.
5. Territory assignment.
6. Source handling.
7. Security for a sensitive population.
8. A process where people regularly ask why something happened.

Then trace the hidden objects behind it.

Which rules touch the process? Which workflows move it? Which assignment lists surface it? Which permissions govern it? Which old decision shaped it? Which report depends on it? Which staff action follows from it?

That trace turns invisible logic into a reviewable map.

The Operating Brief

The practical resource for this chapter is an operating brief.

Use it for one hidden object or one connected process. It should be short enough for someone to read during actual work. One page is often enough.

For a hidden object, answer these questions:

1. What is the object?
2. What type of object is it?
3. What does it do in plain language?
4. What triggers it?
5. Which population or records does it affect?
6. What process depends on it?
7. What depends on its output?
8. Who owns the business decision?
9. Who understands the technical build?
10. How should it be tested?
11. What happens if it breaks or runs incorrectly?
12. When was it last reviewed?
13. Should it be kept, documented, rebuilt, consolidated, restricted, or retired?

For a connected process, use a slightly wider version:

1. What process is being traced?
2. What visible object starts the process?
3. Which hidden logic acts on the record?
4. Which staff role sees the output?
5. Which action should happen next?
6. Which report or review checks the outcome?

7. Which objects are high risk?
8. Which objects need documentation?
9. Which objects should be retired or rebuilt?
10. Who owns the next review?

The operating brief prevents a common failure. A team fixes the object without capturing the reason. The next person inherits a cleaner object with a weak memory trail.

Good documentation gives future staff a path back into the decision.

Retirement Is Governance

Slate teams are often better at building than retiring.

A new need appears, so a new object gets created. A new cycle begins, so a new query gets added. A new process emerges, so a new rule handles it. A new campaign launches, so a new population supports it.

Over time, the instance fills with objects that were useful once. Some still are. Some have become harmless clutter. Some still shape work in ways the team no longer intends.

Governance has to include retirement.

Retiring an object can mean the object did its job. It can mean the institution changed. It can mean a better structure exists. It can mean the process moved elsewhere. It can mean a workaround can finally become a standard.

Before retiring an object, ask:

1. What uses this object today?
2. Which reports, queries, rules, workflows, campaigns, portals, or users depend on it?
3. What happens if it stops running?
4. Which historical records need preservation?
5. Who needs to know before it changes?
6. What date should the retirement occur?
7. What evidence will confirm that retirement was safe?

Some objects should be marked as inactive with notes. Some should be renamed before retirement. Some need a replacement first. Some should stay active with a review date.

The craft is knowing when a tool has served its purpose.

What Teams Should Do Next

Start with one object your team would struggle to explain.

Choose a rule, workflow, assignment list, permission group, legacy report, source handling process, or quiet automation. Avoid starting with the whole instance. Choose one object that still shapes daily work.

Begin by explaining it before changing it.

Write the operating brief. Capture what the object does, why it exists, who owns it, who is affected, what happens if it breaks, when it was last reviewed, and whether it should be kept, rebuilt, restricted, or retired.

Then share the brief with the people who live with the process. A builder may understand the configuration. Staff may understand the behavior. A manager may understand the decision. Security or data owners may understand the risk. The object becomes clearer when those views meet.

Then decide one action.

Maybe the object needs a better name. Maybe it needs a test record. Maybe it needs an owner. Maybe it needs documentation. Maybe it needs retirement. Maybe it is working well and simply needs a review date.

Any of those outcomes is progress.

Repeat the process where the risk is highest. High risk objects deserve review cadence. Lower risk objects need enough documentation to keep them explainable.

The hidden operating system gets safer when it becomes visible.

How To Study The Featured Sessions

Study the Carnegie and Rockhurst session for hidden rule awareness. Focus on how rules power visible behavior and why rule inventories need purpose, ownership, trigger logic, and review.

Study the Technolutions assignment lists and cadences session for work routing. Pay attention to how lists and cadences shape daily staff action, workload expectations, and follow up rhythm.

Study the Technolutions universal workflows session for process modernization. Treat workflow migration as a chance to recover the current process, clarify statuses, improve naming, and align configuration with real staff work.

Study the Harvey Mudd and Huron session for legacy decision excavation. Use the frame to recover why objects exist before deciding whether to keep, rebuild, or retire them.

Study the Asbury, Miami University of Ohio Advancement, and Baylor security session for permission governance. Read access as part of the operating system, with special attention to role change, export power, sensitive data, broad groups, and review cadence.

Read all five sessions with one question in mind:

Which hidden decision is shaping visible work?

That question turns a set of technical topics into one stewardship practice.

Closing

The riskiest object is the one everyone forgot exists.

The point is simple. Hidden logic shapes staff work every day, and important hidden logic deserves ownership.

Rules, workflows, assignment lists, permissions, and legacy decisions are the operating system of the Slate instance.

When that system is explainable, the institution can improve it with confidence. When it is invisible, every change carries more risk.

The practical move is small:

Pick one hidden object and write its operating brief.

Object. Type. Trigger. Population. Owner. Last reviewed. Risk if broken. Documentation link. Future plan.

That is enough to begin.

The hidden operating system gets better one visible decision at a time.

Source Notes

1. "Behind the Music: The Hidden Rules Powering Your Instance," Nina Lifshey, Darian Quinn, and Drew Griffin, Carnegie and Rockhurst University. This session frames rules as the hidden logic that powers visible Slate behavior and shows why rule inventories, owners, triggers, and review habits matter.
2. "Assignment Lists & Cadences," Traci Giovinazzo, Colby Vere, and Benjamin Szabo, Technolutions. This session shows how assignment structures and cadences route work, shape staff expectations, and require clear logic, action definitions, and review ownership.
3. "Workflows: From Legacy to Universal," Ken Higgins, Technolutions. This session frames workflow modernization as more than object conversion. It is a chance to recover process intent, clarify status meaning, and rebuild around current staff practice.
4. "Written in Slate (Not Stone): Excavating Legacy Decisions and Building for Scale," Raissa Diamante and Erin Gore, Harvey Mudd College and Huron. This session gives teams a useful

posture for inherited systems: recover the reason behind old decisions before choosing to keep, rebuild, or retire them.
5. "Don't Forget to Lock the Doors: Enhancing Security and Data Protection," Nathan Vick, Emma Ortiz, Amy Shaiman, and Matthew Haugen, Asbury University, Miami University of Ohio Advancement, and Baylor University Undergraduate Admissions. This session places permissions and data protection inside everyday Slate governance, with emphasis on role based access, review cadence, and documented exceptions.

Chapter 7 - The Build Is Not Done Until It Can Be Taught

Completion is not competence.

A Slate build can be live. It can work. It can solve the immediate problem. It can look strong in a demo. Staff may like it. Leaders may approve it. The first week may go well.

The work may still be unfinished.

If one person understands the build and everyone else depends on that person to explain, repair, approve, or maintain it, the institution has created dependency. The system may function. The knowledge around it is too narrow. That is a fragile kind of success.

Advanced Slate work needs a higher standard.

A build is complete when the right people can operate it without guessing. The owner can explain why it exists. Another person can maintain it. Staff can use it correctly. Documentation exists in a form people can actually use. Training has moved beyond a single walkthrough. The team knows what to do when something changes.

That standard uses role fit. A counselor needs different knowledge from an operations lead. A department administrator needs different technical depth from a central Slate administrator. A seasonal reader needs a smaller dependency map than a Reader process owner.

The standard is role fit.

The right people need the right level of understanding for the work they are expected to use, maintain, explain, or govern.

This chapter draws from five Slate Summit sessions:

1. "Hitting the Right Notes: Designing a Scalable User Training Program," presented by Brady Alumbaugh from Purdue University Undergraduate Admissions.
2. "This Is Not a Side Quest: Gamifying Training Using Interactive Forms," presented by Katie Jordan and Tommy Kennedy from The Parish Group and Villanova University Graduate Admissions.
3. "Durable Change By Design," presented by Ben Parsons and Sarah Kotlinski from College of the Holy Cross and Gettysburg College.
4. "Written in Slate (Not Stone): Excavating Legacy Decisions and Building for Scale," presented by Raissa Diamante and Erin Gore from Harvey Mudd College and Huron.
5. "In Perfect Harmony: Centering Department Administrators in Decentralized Graduate Admissions," presented by Emily Jiang from Stanford University Graduate Admissions.

The sessions approach training from different directions: scalable programs, interactive practice, change adoption, legacy documentation, and decentralized administration.

Together, they point toward one lesson:

Slate knowledge transfers through design.

It transfers through structure, practice, documentation, role clarity, repetition, and review.

The practical question is direct:

How can Slate teams make important builds teachable enough to survive daily use, staff change, and future improvement?

Training Is Part Of Architecture

Training is often treated as the last step in a project. The build is finished. The builder schedules a session. Staff attend. The recording is posted. A document may be shared. The project closes.

That approach can work for simple changes. It fails when the build changes staff behavior, relies on data definitions, affects student experience, introduces maintenance needs, or carries operational risk.

Complex Slate work needs training built into the architecture.

If a workflow changes how records move, training has to explain the process agreement behind the workflow. If a campaign changes student communication, training has to explain audiences, suppressions, content ownership, and review. If a portal changes what students see, training has to explain the user experience and maintenance model. If a report shapes a leadership decision, training has to explain definitions, row grain, limits, and proper use.

The training is part of the build because the build changes how people act.

When training is thin, adoption becomes guesswork. Staff may use the tool inconsistently. They may follow the first example too literally. They may trust an output they cannot interpret. They may avoid the tool because they are unsure what it means. They may return every question to the original builder.

That pattern usually reflects a design gap.

The training missed the work.

A good project plan should include the teaching plan from the beginning. The team should know who needs to learn, what they need to do, what they need to understand, what practice they will complete, what documentation will support them, and how the team will confirm that knowledge moved.

The build and the teaching plan belong together.

Training Should Follow Responsibility

Brady Alumbaugh's "Hitting the Right Notes" is a useful anchor because it starts from a scalable training mindset.

Scalable training serves different roles without starting over each time.

An admissions counselor, operations coordinator, graduate program administrator, Deliver builder, Reader manager, portal maintainer, and senior Slate administrator all need different paths. They still need some shared orientation. They need common vocabulary. They need to know how the institution uses Slate. They need boundaries. They need a place to find documentation. They need to know which actions are theirs and which actions require support.

The first design move is to connect training to responsibility.

A user who completes a process needs operational competence. A process owner needs enough knowledge to explain purpose, review performance, and request changes responsibly. A maintainer needs enough technical understanding to update approved parts safely. A governance owner needs enough context to ask good questions. A builder needs technical depth and system awareness.

Those are different thresholds.

Many training programs struggle because they blur those thresholds. Users receive details they will never use. Maintainers receive instructions that stop at basic operation. Leaders see reports without enough context to interpret them. Department users inherit local responsibility without a clear support model.

Role based learning paths solve part of that problem.

The same source material can support several paths. A campaign overview might serve all users. The audience logic might serve the owner and maintainer. The Liquid details might serve the builder. The approval checklist might serve the governance group. The reporting note might serve leadership.

The structure matters more than the volume of material.

For each role, the training should answer four questions:

1. What does this person need to do?
2. What does this person need to understand?
3. What should this person avoid changing?
4. When should this person ask for help?

Those questions keep training aligned with real work.

Practice Makes Understanding Visible

Katie Jordan and Tommy Kennedy's "This Is Not a Side Quest" adds an important point: training has to be active.

Slate knowledge is procedural. People learn by making decisions, testing records, interpreting results, seeing edge cases, and connecting a screen action to a process purpose.

A passive walkthrough rarely gives enough evidence of learning.

Interactive forms, short scenarios, guided practice, and light game mechanics can make training more useful because they ask learners to do something. The learner chooses a path, diagnoses a record, identifies a correct next step, explains a status, or completes a task. The trainer can see where the logic landed and where confusion remains.

The point is practice.

A training activity can be simple:

1. After Reader training, ask the learner to identify which records are ready for review and which records need attention first.
2. After event training, ask the learner to choose the next action for an attendee, a no show, and a student with incomplete registration details.
3. After Deliver training, ask the learner to identify the audience, suppressions, and expected result for a message.
4. After portal training, ask the learner to find the next step for three student scenarios.
5. After reporting training, ask the learner to explain what a dashboard can support and which decisions need other evidence.

These exercises make learning visible.

Attendance at a training session says little. A watched recording says little. A completed slide deck says little. Competence appears when someone can perform the task, explain the logic, recognize risk, and know when to escalate.

That is why active training is worth the time. It gives the team evidence.

When a learner struggles, the training has produced useful information. The gap might be a confusing document, a missing definition, a weak example, an unclear process, or an assumption held by the builder. The team can improve the artifact because the practice exposed the gap.

Active training turns confusion into design feedback.

Change Needs A Routine

Ben Parsons and Sarah Kotlinski's "Durable Change By Design" broadens the conversation from training sessions to adoption.

Many Slate projects succeed at launch and weaken later. The original build works. The first users understand it. The energy is real. Then staff change, priorities shift, edge cases appear, documentation ages, and the process starts to drift.

Durable change requires design beyond launch.

The team needs an owner, a routine, and a way for the process to survive staff change.

The owner keeps the process connected to institutional purpose. The routine makes the process part of normal work. The survival plan protects the work when people move roles, leave the office, or lose proximity to the original project.

For a workflow, that may mean a monthly review of stuck records, a status definition guide, and a named process owner. For a campaign, it may mean an annual content review, a pre send QA checklist, and an owner for each audience. For a portal, it may mean a page owner, a term based review date, and a staff guide for common questions. For a report, it may mean a definition note, a meeting routine, and a clear statement of intended use.

The adoption question should be part of project planning:

How will this become normal work?

That question changes the build. It forces the team to think about staff behavior, documentation, ownership, review cadence, and feedback loops. It also keeps the team from treating launch as the finish line.

Change becomes durable when the institution knows how to carry it.

Documentation Should Preserve The Why

Raissa Diamante and Erin Gore's "Written in Slate (Not Stone)" connects training to institutional memory.

Every Slate instance carries history. Some of that history is visible in names, folders, rules, reports, workflows, queries, prompts, source formats, and portal pages. Some of it lived in meetings and hallway decisions. A staff member solved a problem during implementation. A workaround helped the office through a deadline. A report supported a leader who later changed roles. A population rule answered a temporary need.

The object remains after the reason fades.

That creates a training problem. New staff learn what exists without learning why it exists. Builders inherit objects without context. Owners hesitate to change processes because the old logic feels

mysterious. Teams either leave old decisions untouched or rebuild them without understanding what they protected.

Legacy documentation can become a teaching tool.

A useful legacy decision log captures the object, the original purpose if known, the current purpose, the population affected, the process it supports, related objects, current owner, risk if changed, and next action.

That kind of documentation teaches new staff how the instance evolved. It separates active decisions from historical residue. It helps teams identify which objects should stay, which should be rebuilt, and which should retire.

The strongest documentation answers five questions:

1. What is this?
2. Why does it exist?
3. How does it work?
4. Who owns it?
5. What should change trigger?

Click instructions answer only part of the need. Staff need to understand purpose and risk as well as steps.

Short documentation can be excellent. A clear operating brief, checklist, short video, practice task, and review date can be more useful than a long guide nobody opens. The test is whether the artifact helps someone do the work, understand the purpose, and know when to ask for help.

Documentation becomes part of training when it preserves the why.

Decentralized Users Need A Shared Model

Emily Jiang's "In Perfect Harmony" brings the training question into decentralized graduate admissions.

Department administrators often carry real process responsibility inside shared institutional systems. They may manage local program needs, review steps, communications, decision support, applicant questions, and reporting expectations. The central team needs standards and security. Local teams need enough autonomy to serve their programs well.

That is where role based knowledge transfer becomes critical.

Decentralized users need clear permissions, clear responsibilities, clear support channels, and clear standards. They need to know what they can change, what they should leave alone, where local variation is allowed, where institutional rules apply, and how to raise an issue.

Generic training will miss the point.

A department administrator may need training on reviewing applicants, managing program specific communication, interpreting status changes, reading reports, and escalating technical issues. A central Slate team may need a parallel habit: listening to local needs, documenting standards, publishing support paths, and designing processes that reduce endless personal help.

The goal is coordinated difference.

Local users should have enough autonomy to do meaningful work. The shared system should have enough structure to keep student experience, data quality, security, and reporting coherent.

Training sits between those needs.

It gives local users confidence without handing them unclear risk. It gives central teams a way to support variation without rebuilding the same explanation every week. It also makes standards feel less arbitrary because users can see the reason behind them.

The Slate Teaching Brief

A teachable build needs a small set of supporting artifacts. The most useful starting point is a Slate teaching brief.

A Slate teaching brief is a short document that connects a build to purpose, role, practice, risk, and maintenance.

It should be specific. "How to use Slate" is too broad. "How our admitted student event follow up process works" is useful. "How to update the application completion campaign safely" is useful. "How department administrators should review and escalate applicant records" is useful.

The brief should cover:

1. Purpose: what the process accomplishes and why it matters.
2. Audience: who uses it, owns it, maintains it, and approves changes.
3. Data: where the key data comes from and which definitions matter.
4. Logic: the major rules, workflows, populations, statuses, or dependencies.
5. Normal use: the steps users follow in ordinary cases.
6. Exceptions: common edge cases and what to do with them.
7. Risk: what could go wrong and which changes need review.
8. Practice: a task or scenario the learner must complete.
9. Teach back: what the learner should be able to explain or perform.
10. Review: owner, review date, and update trigger.

This format keeps the artifact compact while avoiding the common mistake of documenting clicks without purpose.

The brief also helps experts translate. Expert builders often forget how much they know. They know which object matters, which field is safe, which query is old, which status name is misleading, and which button carries risk. New users need that mental map made visible.

Good training makes the map visible.

Instead of saying, "Run the report," the brief explains what the report means and what decision it can support. Instead of saying, "Update the population," the brief explains which population, why it exists, what it affects, and what changes need review. Instead of saying, "Use this workflow," the brief explains which statuses matter and when to escalate.

That translation is one mark of a strong Slate culture.

Experts make the system legible.

Training Levels Help Teams See Risk

Every important process should have a known competence level attached to the people around it.

A simple five level model works well.

1. Orientation. The person understands what the process is, why it exists, and where to find help.
2. Operation. The person can perform the process correctly in normal cases.
3. Troubleshooting. The person can recognize common errors and know what to check.
4. Maintenance. The person can update approved parts of the process safely.
5. Stewardship. The person can evaluate whether the process still fits the institution's needs and guide future changes.

Every important process needs coverage at the right level.

This model helps teams see concentrated risk. One person may be at level five for too many processes while everyone else sits at level one or two. The office may function day to day. The risk still sits in one person.

The solution is intentional transfer.

Start with high use and high risk processes: application review, event management, communications governance, portal updates, permission management, assignment lists, source handling, reporting definitions, rules that affect student outcomes, and workflows that move records.

Choose one process. Identify the current level of understanding across users, owners, maintainers, and stewards. Then decide which gap matters most.

The team may need a user checklist. It may need a maintainer guide. It may need a stewardship conversation. It may need a second person trained at level four. The level model helps the team choose the right artifact.

Teach Back Turns Training Into Evidence

A teach back is simple. After training, ask the learner to explain the process and complete a relevant task.

This should feel like practice and verification. The purpose is to see whether knowledge moved.

The teach back can ask:

1. Can the person explain the purpose of the process?
2. Can the person identify where the data comes from?
3. Can the person describe the major logic?
4. Can the person perform a basic task correctly?
5. Can the person identify what could go wrong?
6. Can the person name the owner or escalation path?
7. Can the person document what changed?
8. Can the person teach another user the basic workflow?

If the learner can do those things at the expected level, the transfer worked. If they struggle, the training has produced a roadmap. The team now knows what definition, example, practice task, or documentation gap needs attention.

This makes teach back one of the most useful habits a Slate team can adopt.

It protects against false completion. A project can launch, training can happen, and adoption can still be thin. Teach back gives the team a way to test whether the system has become understandable enough to carry forward.

Documentation Needs Ownership

Documentation ages unless someone owns it.

A process changes. A field is renamed. A workflow is rebuilt. A screenshot becomes outdated. A campaign retires. A source format shifts. A new user follows an old instruction and gets confused.

The answer is documentation maintenance.

Every important training artifact should have an owner and review date. The review can stay light. The owner should check whether the process is current, screenshots are accurate, links still work, owners are correct, examples still match current practice, and the artifact still helps the intended role.

Review triggers matter too. A training artifact should be reviewed when:

1. The workflow changes.
2. The population logic changes.
3. A field, prompt, source, or status changes.
4. The owner changes.
5. A related report changes.
6. A staff member using the artifact gets confused.
7. A cycle closes and the process is due for review.

This turns documentation from a static file into part of the operating system.

When documentation has ownership, training becomes easier to trust. Staff know where to look. Owners know what to update. Builders know which artifact needs revision after a change. Leaders can see that training is connected to maintenance rather than scattered across old folders.

What Teams Should Do Next

Pick one advanced process and ask someone else to explain it.

Choose someone who uses, owns, supports, or depends on the process. Avoid choosing the original builder first. The point is to see how much knowledge has moved beyond the builder.

Ask what the process is for. Ask where the data comes from. Ask what logic matters. Ask what could go wrong. Ask what can be changed safely. Ask who owns the process. Ask where documentation lives. Ask what they would do if something looked wrong.

Listen for gaps.

The gaps are useful. They show where the training needs to be clearer, where documentation is missing, where expert assumptions are hiding, and where the next practice task should be built.

Then create one artifact.

Make it small enough to finish. A one page operating brief. A five minute walkthrough. A checklist. A practice form. A safe exercise. A teach back rubric.

Use it with one learner. Watch where they pause. Improve the artifact. Assign an owner. Set a review date.

That is enough to begin.

How To Study The Featured Sessions

Study the Purdue session for scalable training structure. Focus on learning paths, role differences, shared orientation, documentation access, and repeatable support.

Study The Parish Group and Villanova session for active practice. Pay attention to how interactive forms can turn training into scenario based evidence.

Study the Holy Cross and Gettysburg session for adoption design. Use it to connect launch, ownership, routine, communication, feedback, and staff transition.

Study the Harvey Mudd and Huron session for legacy knowledge transfer. Treat legacy documentation as a way to preserve institutional memory and teach why old objects exist.

Study the Stanford session for decentralized administration. Focus on how central standards, local autonomy, support paths, and role clarity can fit inside one shared operating model.

Read all five sessions with one question in mind:

Who needs to understand this build well enough to carry it forward?

That question turns training from a late project task into a design requirement.

Closing

Completion is not competence.

Advanced Slate work is finished when the institution can carry it. The right people can use it, explain it, maintain it, document it, and teach it forward.

That standard changes the work. Every meaningful build should create more than a working object. It should create transferable understanding.

The practical move is small:

Pick one process that depends too much on one person.

Ask someone else to explain it.

Use the gaps to create one teaching artifact.

Then give that artifact an owner and a review date.

The build is done when the institution can teach it.

Source Notes

1. "Hitting the Right Notes: Designing a Scalable User Training Program," Brady Alumbaugh, Purdue University Undergraduate Admissions. This session frames Slate training as a scalable program,

with role based learning paths, common orientation, documentation access, and repeatable support.
2. "This Is Not a Side Quest: Gamifying Training Using Interactive Forms," Katie Jordan and Tommy Kennedy, The Parish Group and Villanova University Graduate Admissions. This session shows how forms, scenarios, and game inspired practice can make learner understanding visible.
3. "Durable Change By Design," Ben Parsons and Sarah Kotlinski, College of the Holy Cross and Gettysburg College. This session places training inside adoption design, with attention to ownership, routines, feedback, communication, and staff transition.
4. "Written in Slate (Not Stone): Excavating Legacy Decisions and Building for Scale," Raissa Diamante and Erin Gore, Harvey Mudd College and Huron. This session connects documentation to institutional memory, especially the reasons behind older objects and inherited processes.
5. "In Perfect Harmony: Centering Department Administrators in Decentralized Graduate Admissions," Emily Jiang, Stanford University Graduate Admissions. This session shows why decentralized users need autonomy, support paths, standards, and role clarity inside a shared Slate model.

Chapter 8 - The Next Slate Practitioner Stack

The next Slate builder is a translator.

Technical skill still matters. Query skill matters. Rules matter. Deliver, Reader, portals, datasets, events, workflows, and reports all matter. The person who knows where the setting lives still has real value.

The work now asks for a wider profile.

Advanced Slate practitioners are translating business problems into system design. They are translating data structures into decisions. They are translating AI output into reviewed work. They are translating local office needs into shared institutional patterns. They are translating clever builds into documentation, training, ownership, and review.

That is the practitioner stack emerging from Slate Summit 2026.

It is a stack because the work rarely sits in one module. A reporting question can depend on query literacy, row grain, source confidence, event design, communication logic, and leadership use. A portal project can depend on content strategy, permissions, datasets, student behavior, testing, and maintenance. An AI experiment can depend on policy, source material, human review, accessibility, and documentation.

The next advanced practitioner will be measured by feature fluency plus the ability to connect features into systems that people can trust, use, maintain, and improve.

This chapter draws from eight Slate Summit sessions:

1. "New Feature Spotlight: Slate AI," presented by Shawn Kelly from Technolutions.
2. "The Captain and the Sidekick: Using Slate AI Without Losing Your Superpowers," presented by Tommy Kroll from Columbia College Chicago.
3. "Vibing with Slate AI: From Prompts to Production Ready Portals," presented by Michael Anzalone from Brown University Graduate School.
4. "Leveraging the Slate AI Assistant: Power, Pitfalls, and Potential," presented by Chris Kizer from Duke University Graduate and Professional Programs.
5. "From Scratch: Key Ingredients for Mixing Custom Datasets Southern Style," presented by Brittany Richardson, Laura Sturgill, and Domenick Rozzi from the University of Alabama, West Chester University, and Huron Consulting Group.
6. "Departments and Dragons," presented by Heather Lefebvre from MIT Office of Graduate Education and Ryan Walsh from Carnegie.
7. "One Slate to Rule Them All," presented by David Mariott, Jake Murphy, and Erin Toney from Oklahoma State University Undergraduate Admissions and Oklahoma State University Institute of Technology.
8. "The Moonshot," presented by Laura MacBride, Dan Hamrin, and Nicolle Skoien from Marquette University.

The sessions cover AI, custom datasets, department support, shared Slate governance, institutional coordination, and ambitious system change.

Together, they point toward one lesson:

Slate work is becoming broader, more connected, and more consequential.

The practical question is direct:

What kind of practitioner does this work now require?

The Stack Is A Map

The stack is a development map. It helps practitioners and teams name the skill areas that advanced Slate work now draws together.

The stack has nine parts:

1. Query literacy.
2. Data architecture.
3. Reporting and action design.
4. Communications governance.
5. Portal and event architecture.
6. Rules and workflow governance.
7. AI evaluation.
8. Training and documentation.
9. Cross unit strategy.

An individual can grow across these areas over time. A healthy team spreads them across roles, owners, and development paths.

The value of the stack is visibility.

It helps one practitioner see where they are growing. It helps a manager see where knowledge is concentrated. It helps a team see which skills are present, which skills are missing, and which skills should be developed next. It also helps leaders understand that advanced Slate work is a professional discipline with durable methods.

The stack also protects teams from the most common mistake in staff development: treating every gap as a training topic.

Some gaps need training. Some need better documentation. Some need standards. Some need governance. Some need a second owner. Some need a smaller project. Some need clearer expectations about who should make which decision.

The practitioner stack helps sort those needs.

AI Evaluation

AI evaluation is the first part of the stack because AI is moving closer to daily Slate work.

Shawn Kelly's "New Feature Spotlight: Slate AI" gives the product view. AI is becoming part of the Slate environment itself. That shift matters because proximity increases responsibility. When AI sits closer to records, copy, portals, prompts, and staff workflow, teams need clear standards for use, review, security, source material, and final ownership.

Tommy Kroll's "The Captain and the Sidekick" gives a useful operating posture. AI can assist. The practitioner still owns the work.

That posture keeps the tool in the right place.

AI can help brainstorm, summarize, draft, reformat, compare, explain, and accelerate. It can help produce a first draft of documentation. It can help explain Liquid. It can help turn messy notes into a project plan. It can suggest a checklist. It can help diagnose a pattern in an error message or outline a portal structure.

The practitioner supplies context, verifies output, checks policy, protects data, tests the result, and decides what belongs in production.

Michael Anzalone's "Vibing with Slate AI" pushes the point into production work. A prompt can produce something interesting. Production work requires a higher bar. It needs source truth, testing, accessibility checks, maintainability, fallbacks, review, and a person who can explain the final result.

Chris Kizer's "Leveraging the Slate AI Assistant" sits in the same practical space. The value of AI is assistance inside a disciplined workflow.

The skill is evaluation.

Can the practitioner define the task clearly? Can they give the model the right context? Can they recognize invented facts? Can they identify sensitive data boundaries? Can they verify the answer against policy or source material? Can they decide when the output is strong enough to revise and when it should be discarded? Can they document how AI was used?

That is a different skill from enthusiasm.

AI fluency in Slate should be measured by judgment. The strongest practitioner uses AI with purpose. They know where AI helps, where it creates risk, and what human review must protect.

Data Architecture

Data architecture is the second part of the stack because advanced Slate work often begins before the build.

"From Scratch" frames custom datasets as more than a technical feature. A dataset is a design choice about how information should be represented, governed, reused, and maintained.

That is a major shift.

Early Slate work often starts with fields, forms, queries, exports, reports, and records. As teams mature, they encounter problems that resist a single field. They need to represent relationships, repeating values, program settings, multi step decisions, external context, reusable configurations, staff assignments, or cycle specific logic.

That is where data modeling enters the work.

A strong practitioner knows when a field is enough. They know when a dataset is the better shape. They know when related records belong in the design. They know when a custom structure needs governance. They know when a simpler approach will serve the office better.

Those are technical questions and institutional questions at the same time.

The practitioner should ask:

1. What real world thing does this data represent?
2. Who owns the values?
3. Who can change them?
4. How often do they change?
5. Which processes depend on them?
6. What happens when the value is missing?
7. Which reports, rules, portals, or campaigns will use the structure?
8. What naming standard and documentation will make it understandable later?

Bad data structure creates future drag. Good data structure creates future leverage.

A messy structure can make every later query harder. It can make reports unreliable. It can make rules fragile. It can make maintenance risky. A strong structure can simplify future work because it gives builders stable material.

Data architecture is the skill of designing that material with care.

Query Literacy

Query literacy remains part of the stack because query skill is the connective tissue between data and action.

A practitioner who can build a query can answer a question. A practitioner with query literacy can examine the question itself.

They ask what population is being described. They ask what row grain is required. They ask which joins are safe. They ask whether the answer should be one row per person, one row per application, one row per event registration, one row per checklist item, or one row per relationship. They ask which filters express the business rule and which filters are only a convenience.

That level of literacy protects the team from false clarity.

A clean looking report can still answer the wrong question. A campaign population can still include the wrong students. A dashboard can still mix grains. A leadership number can still look precise while hiding a definition problem.

The next practitioner needs enough query literacy to challenge the shape of the answer.

The point is practical. Teams need shared language around population, grain, joins, subqueries, reuse, and governance. Shared language lets more people participate in defining better questions.

Query literacy also connects to other parts of the stack. Communications governance needs trustworthy audiences. Event signal design needs clear attendance and registration definitions. AI evaluation needs source material that means what it says. Reporting and action design needs grain clarity. Data architecture needs a way to test whether the structure can answer real questions.

The query is often where the hidden assumptions become visible.

Reporting And Action Design

Reporting and action design is the skill of connecting evidence to use.

A dashboard, counselor list, campaign population, workflow queue, and operational report may share data. They still serve different purposes.

A dashboard may help leaders see trend, risk, and exception. A counselor list may need next steps at the student level. A campaign population may need audience rules and suppressions. A workflow queue may need status clarity and ownership. An operational report may need refresh cadence, accountability, and review routine.

The practitioner has to translate the request.

"We need a report" is rarely the full need. The deeper need is usually a decision, a meeting routine, a staff action, a risk check, a communication audience, or a process review.

Strong reporting design starts with use:

1. Who will use this?

2. What decision or action should it support?
3. What evidence is needed?
4. What level of detail is useful?
5. How fresh must the data be?
6. What definitions need to be visible?
7. What should the user avoid concluding from it?
8. Who owns review and maintenance?

Those questions keep reports from becoming decorative.

They also make reports easier to teach. A report with an owner, decision, definition note, refresh cadence, and interpretation guide can become part of office practice. A report missing that context can become another tab people open and misunderstand.

The next practitioner knows that information alone is incomplete. The question is what the information should help people do.

Communications Governance

Communications governance belongs in the stack because Slate communications have expanded beyond message delivery.

They are audience logic, content strategy, source confidence, timing, personalization, suppression, review, testing, and measurement in one system.

A practitioner working in Deliver has to think like a builder and an editor. They need to understand query logic, Liquid, populations, content blocks, campaign timing, opt outs, source data, and student experience. They also need judgment about when personalization helps and when it creates fragility.

The governed personalization chapter treated this in detail. The practitioner stack adds the development question: who on the team can design communication systems responsibly?

That skill includes several moves:

1. Define the audience in plain language before building the query.
2. Name the purpose of the message.
3. Identify the source fields that drive personalization.
4. Decide which content should be reusable.
5. Define suppressions before launch.
6. Test the message against real scenarios.
7. Assign owners for audience, content, logic, and review.
8. Measure the result in a way that fits the decision.

The practitioner should know when to simplify.

If one message requires too many branches, the issue may be strategy rather than Liquid. If an audience definition takes too much explanation, the process may need clearer categories. If personalization depends on weak data, the message should avoid using that data as truth.

Communications governance is the skill of making messages relevant while keeping the system maintainable.

Portal And Event Architecture

Portal and event architecture is the skill of designing student experience and operational evidence together.

Events create structured moments of engagement. Portals create structured spaces for action, information, and return behavior. Both can support the student journey. Both can produce signals. Both can become messy if the design lacks purpose.

The advanced practitioner needs to match complexity to use.

A major admitted student event may justify schedule logic, audience segments, post event follow up, family content, department routing, and signal review. A small information session may need a clear registration form, reliable attendance tracking, and one clean follow up path. A portal may be the right answer when it organizes next steps. It may create extra maintenance if it only repeats content available elsewhere.

The practitioner should ask:

1. What student action should this experience support?
2. What staff action should it support?
3. What data enters the system?
4. What signal should be captured?
5. What follow up should happen?
6. What maintenance will the experience require?
7. What will be reviewed after the cycle?

That puts portals and events inside one design frame.

The design is stronger when the event and portal tell one operational story. A student registers, attends, returns to the portal, views aid information, signs up for a department session, and completes a next step. Each action may mean something different. The practitioner has to avoid overstating any one signal while still using the pattern to support better outreach.

Portal and event architecture is the skill of designing the experience and the evidence at the same time.

Rules And Workflow Governance

Rules, workflows, assignment lists, permissions, templates, and inherited objects form the hidden operating system of a Slate instance.

The advanced practitioner needs to know how to build them and how to govern them.

A rule should have a purpose. A workflow should match a real process. An assignment list should carry an action expectation. A permission group should reflect role and risk. A legacy object should have context. A source format should have an owner. A query that runs every week should have a reason to exist.

This skill matters because hidden logic can shape student experience and staff behavior while staying out of view.

The practitioner should ask:

1. What process does this object represent?
2. Who owns it?
3. What depends on it?
4. What happens if it fails?
5. What should be tested before changing it?
6. How will someone know whether it is still current?
7. What documentation makes the logic understandable?

These questions turn technical objects into governed objects.

Rules and workflow governance also protects future builders. A future staff member should be able to understand why a rule exists, what it touches, and what review is needed before change. That context helps the office clean up with care.

The next practitioner treats hidden logic as shared infrastructure.

Training And Documentation

Training and documentation are part of the practitioner stack because advanced Slate work has to move beyond one expert.

The previous chapter argued that the build is complete when the institution can teach it. That standard belongs here too.

A practitioner who creates a useful build should also create transferable understanding. That might mean an operating brief, source map, test case, practice task, short walkthrough, owner note, change log, or teach back prompt.

Documentation should answer more than where to click.

It should explain what the object is, why it exists, how it works, who owns it, what depends on it, what can go wrong, and what change should trigger review.

Training should follow responsibility. A counselor needs a different level of understanding from a central Slate administrator. A department user needs a different support model from a communications owner. A leader needs interpretation guidance rather than builder detail.

The practitioner has to translate the system for each role.

That translation is part of the work. It protects against dependency, drift, and avoidable support loops. It also helps teams grow future practitioners rather than relying on one person to hold the system together.

Training and documentation are how technical skill becomes institutional skill.

Cross Unit Strategy

Cross unit strategy is the skill of helping multiple offices use one system coherently.

"Departments and Dragons," "One Slate to Rule Them All," and "The Moonshot" point toward this larger challenge.

Departments, programs, schools, graduate units, professional programs, undergraduate admissions, central enrollment, and institutional partners may all have real needs. Their timelines, language, review habits, communication expectations, reports, and local processes can differ.

The practitioner preserves meaningful difference while preventing each unit from creating an isolated version of reality.

The work is coordinated flexibility.

Common standards where the institution needs them. Local variation where the work justifies it. Clear support paths. Shared vocabulary. Permission boundaries. Ownership maps. Review routines. A way to decide what belongs centrally and what belongs locally.

That requires translation.

The practitioner listens to a local request and identifies the real need. It may call for configuration. It may call for training. It may call for documentation. It may call for a governance conversation. It may call for a policy decision. It may call for a clear refusal with a clear reason.

"The Moonshot" adds another layer: ambitious Slate work needs sequencing and ownership. Major institutional projects require more than technical confidence. They need a path from current practice to future practice. They need stakeholder alignment, communication, testing, training, and maintenance.

Cross unit strategy is the skill of connecting Slate architecture to institutional behavior.

The Practitioner Stack Self Assessment

The companion resource for this chapter is a Practitioner Stack Self Assessment.

It is a development tool. Each skill area should be rated by current confidence and paired with one artifact that would prove growth.

The artifact matters because interest is cheap. Evidence is better.

For AI evaluation, the artifact might be a prompt review checklist, a safe use policy, or a documented workflow for using AI to draft Slate support material.

For data architecture, the artifact might be a decision guide for fields, datasets, related records, and external sources.

For query literacy, the artifact might be a reusable query pattern that explains population, row grain, joins, and known limits.

For reporting and action design, the artifact might be one report with an owner, decision statement, refresh cadence, and interpretation notes.

For communications governance, the artifact might be a campaign governance map.

For portal and event architecture, the artifact might be an event signal loop or portal tab plan with owner, audience, data dependencies, and review timing.

For rules and workflow governance, the artifact might be an audit of one workflow trigger, including the business process it represents and what should be tested before change.

For training and documentation, the artifact might be a teach back rubric or operating brief for a process someone else needs to learn.

For cross unit strategy, the artifact might be an ownership matrix that shows what is central, what is local, who approves changes, and where support requests go.

This approach keeps development practical.

The practitioner learns the idea, builds a small object, tests it with another person, and improves it. That is how a skill moves from interest to capability.

A One Quarter Development Plan

A one quarter development plan should stay narrow.

Choose one primary skill and one adjacent skill.

If the primary skill is AI evaluation, pair it with documentation. Use AI to draft support material, then build the review process around it.

If the primary skill is data architecture, pair it with reporting. Build a structure that feeds a real decision.

If the primary skill is communications governance, pair it with query literacy. Make sure the population logic supports the message strategy.

If the primary skill is portal architecture, pair it with event follow up. Use the portal to continue the experience after a student engages.

If the primary skill is workflow governance, pair it with training. Teach staff what the workflow statuses mean and how to act.

If the primary skill is cross unit strategy, pair it with permission and ownership mapping. Shared systems need clear boundaries.

Then use a simple three month rhythm:

1. Month one: build one artifact.
2. Month two: test or use the artifact in real work.
3. Month three: teach, document, or revise the artifact based on what happened.

That is enough for a quarter.

The goal is to move one skill from interest to evidence. The artifact gives the team something to review. It also gives the practitioner a cleaner story about their growth.

What Teams Should Do Next

Teams should use the stack to see concentration and gaps.

Start with one high value area of work. Communications, reporting, portals, events, Reader, workflows, AI use, or shared unit support can all work.

Ask which stack skills the work depends on.

Then ask who has those skills today, who needs enough understanding to support the work, and which artifact would help knowledge move.

For example, a communication project may depend on query literacy, communications governance, AI evaluation, and documentation. A portal project may depend on data architecture, portal architecture, rules governance, reporting design, and training. A shared graduate process

may depend on cross unit strategy, permissions, workflow governance, documentation, and local support.

The stack makes that dependency visible.

It also helps leaders avoid vague professional development. "Learn more Slate" is too broad. "Build a campaign governance map and teach the owner how to use it" is concrete. "Improve reporting skill" is too broad. "Create one report that names its decision, owner, grain, refresh cadence, and action path" is concrete.

Development should produce evidence.

That evidence can be small. A checklist. A query pattern. A workflow audit. A source map. A prompt policy. A report note. A portal review plan. A teach back guide. A unit ownership matrix.

The artifact proves that learning moved into practice.

How To Study The Featured Sessions

Study the Slate AI spotlight for the direction of product capability. Focus on what changes when AI becomes closer to the workflow.

Study "The Captain and the Sidekick" for operating posture. Use it to keep the practitioner responsible for judgment, review, and final ownership.

Study "Vibing with Slate AI" for the move from prompt exploration to production work. Pay attention to review, testing, accessibility, and maintainability.

Study "Leveraging the Slate AI Assistant" for practical assistance patterns and the risks that require review.

Study "From Scratch" for data architecture. Focus on the decision between fields, datasets, reusable structures, governance, and future maintenance.

Study "Departments and Dragons" for department support inside shared systems. Watch how local needs, support models, standards, and autonomy interact.

Study "One Slate to Rule Them All" for shared environment coordination. Focus on naming, permissions, ownership, reports, templates, and institutional standards.

Study "The Moonshot" for ambitious system change. Use it to think about sequencing, stakeholder alignment, ownership, and durable adoption.

Read all eight sessions with one question in mind:

Which practitioner skill does this work require, and what artifact would prove growth in that skill?

That question turns Summit inspiration into development.

Closing

The next Slate builder is a translator.

That person stands between the feature and the decision. Between local need and shared standard. Between AI assistance and human accountability. Between the clever build and the maintainable system. Between what one office wants now and what the institution needs to support later.

That role asks for technical skill, systems thinking, judgment, and teaching ability.

The practical move is simple:

Choose one skill from the stack.

Build one artifact that proves growth.

Use it with another person.

Improve it.

That is how practitioners grow and how offices build capacity.

The next Slate practitioner will be defined by the ability to connect pieces into systems people can trust, use, maintain, and improve.

That is the stack.

Source Notes

1. "New Feature Spotlight: Slate AI," Shawn Kelly, Technolutions. This session shows AI becoming part of the Slate product environment and raises the need for review, security, source awareness, and human accountability.
2. "The Captain and the Sidekick: Using Slate AI Without Losing Your Superpowers," Tommy Kroll, Columbia College Chicago. This session offers a practical posture for AI use, with the practitioner owning judgment while AI assists with drafting, brainstorming, explanation, and acceleration.
3. "Vibing with Slate AI: From Prompts to Production Ready Portals," Michael Anzalone, Brown University Graduate School. This session frames the move from prompt exploration to production work that needs testing, accessibility, maintainability, and review.
4. "Leveraging the Slate AI Assistant: Power, Pitfalls, and Potential," Chris Kizer, Duke University Graduate and Professional Programs. This session reinforces AI as assistance inside a disciplined workflow rather than a replacement for practitioner judgment.

5. "From Scratch: Key Ingredients for Mixing Custom Datasets Southern Style," Brittany Richardson, Laura Sturgill, and Domenick Rozzi, University of Alabama, West Chester University, and Huron Consulting Group. This session points toward data architecture as a core Slate skill, especially around reusable structures, ownership, and maintenance.
6. "Departments and Dragons," Heather Lefebvre, MIT Office of Graduate Education, and Ryan Walsh, Carnegie. This session shows the importance of local autonomy, central standards, support paths, and role clarity in department based Slate work.
7. "One Slate to Rule Them All," David Mariott, Jake Murphy, and Erin Toney, Oklahoma State University Undergraduate Admissions and Oklahoma State University Institute of Technology. This session highlights shared environment coordination across institutional contexts, including standards, permissions, naming, templates, reports, and ownership.
8. "The Moonshot," Laura MacBride, Dan Hamrin, and Nicolle Skoien, Marquette University. This session connects ambitious Slate work to sequencing, stakeholder alignment, ownership, adoption, and long term institutional capability.

Chapter 9 - The Work After The Summit

A good conference creates a particular kind of energy.

People come home with session notes, screenshots, new language, promising examples, and a few ideas that felt obvious the moment someone said them. Then the calendar returns. The inbox returns. The same projects are still waiting. The same deadlines still matter. The same decisions still need to be made.

That is where conference learning either becomes practice or becomes a folder.

The first eight chapters in this collection traced patterns across Slate Summit 2026: decision infrastructure, query skill, signal design, communications governance, events and portals, hidden logic, teachable builds, and the next practitioner stack. This chapter has a different job. It translates those ideas into a path a team can use after the Summit ends.

Conference insight becomes valuable when it changes how work is done.

That change can start without a giant backlog. In fact, the giant backlog is often the reason good ideas stall. It collects everything and commits to very little. A stronger approach is smaller and more disciplined. Choose one decision, one process, one signal, one campaign, one hidden object, one teaching artifact, and one practitioner skill. Use those seven choices to turn broad learning into operating practice.

That is the work after the Summit.

Conference Energy Needs A Path

The easiest thing to bring home from a conference is inspiration. The harder thing is translation.

Translation means taking something heard in a session and turning it into a decision, a process, an owner, a build, a training artifact, or a review habit. It means asking what should change on Monday morning, what should be planned for the next cycle, and what should be studied before the office moves.

Slate teams need that translation because the work has expanded.

They are doing more than building forms, reports, campaigns, and portals on request. They are shaping how offices think, how data moves, how messages adapt, how events become evidence, how staff learn systems, and how institutional knowledge survives turnover.

That is real design work.

It needs judgment as much as configuration skill. A session can show what is possible. A local team still has to decide what is appropriate, sustainable, teachable, and worth maintaining in its own setting.

The Summit gives examples. The office has to build practice.

The Series In One Frame

Chapter 1 started with decision infrastructure. Reports, dashboards, queries, campaigns, portals, and workflows should support decisions people actually make. A dashboard that changes no action is decoration. A modest report that clarifies who acts next can be infrastructure. After the Summit, this frame gives teams a filter. When twenty ideas sound promising, ask which one improves a decision that already matters.

Chapter 2 moved into the Query Ladder. Query work is often treated as a technical output: someone asks, someone builds. The stronger view is developmental. A team learns to move from question to base population, from base population to row grain, from row grain to joins, from joins to reusable logic, and from reusable logic to governance. Query expertise should become teachable capacity, rather than private magic held by one person.

Chapter 3 focused on signals and action. Slate produces signal everywhere: clicks, visits, form submissions, portal activity, event attendance, checklist movement, counselor interactions, academic interests, aid engagement, reader outcomes, and more. Signal alone is raw material. It becomes useful when someone defines what it suggests, what it fails to prove, who should see it, and what action it should inform.

Chapter 4 moved that same discipline into personalization. Personalization can make communication more relevant. It can also create a maze of exceptions, copied messages, conditional branches, and mystery rules. Governed personalization uses reusable blocks, audience rules, suppression logic, review habits, ownership structures, and testing. The question is whether the system remains understandable after the clever build is done.

Chapter 5 treated events and portals as signal systems. Events are structured moments of engagement. Portals are structured spaces for action, information, and return behavior. Both can reveal intent, confusion, readiness, concern, or momentum. The work is to notice behavior without overstating it, then connect the pattern to a responsible next step.

Chapter 6 explored the hidden operating system. Every Slate instance has visible objects and invisible logic. Forms, portals, emails, events, prompts, reader screens, and reports are visible. Rules, permissions, lookup tables, source formats, naming patterns, query conventions, inactive automations, and old decisions often sit beneath the surface. That hidden layer shapes what staff can do, what data means, and how future work behaves.

Chapter 7 argued that the build is complete when it can be taught. A working Slate build matters. A working build that another person can understand, test, maintain, and explain matters more. Teaching exposes weak assumptions. It turns private knowledge into shared practice. It protects the office from dependency on one builder.

Chapter 8 named the next Slate practitioner stack. Advanced Slate work now requires more than technical fluency. It draws on data literacy, process design, communications judgment, governance instinct, documentation habits, training ability, cross unit coordination, and AI evaluation. Staff development has to sit inside system development.

Together, these chapters point to one practical conclusion:

Summit ideas become useful when they become operating habits.

The Summit Adoption Model

The adoption model is intentionally small.

Choose one decision, one process, one signal, one campaign, one hidden object, one teaching artifact, and one practitioner skill.

That is enough to create motion without pretending the whole office can be redesigned in a month.

The model works because each choice touches a different part of the system. The decision ties the work to institutional use. The process reveals the workflow behind the request. The signal connects student or staff behavior to evidence. The campaign makes governance visible. The hidden object addresses maintenance. The teaching artifact turns knowledge into team memory. The practitioner skill makes staff growth part of the project.

This is small enough to use and broad enough to matter.

One Decision

Start with a decision your office makes repeatedly.

It could be who gets counselor outreach this week. It could be which admitted students need a different message. It could be which event audience should receive a second invitation. It could be which portal behavior should trigger follow up. It could be which reports are reviewed during a weekly meeting.

Write the decision in plain language.

Then ask:

1. What evidence supports this decision today?
2. Where does that evidence live?
3. Who owns the action?
4. What timing matters?
5. What would change if the evidence were clearer?
6. What should the office avoid concluding from this evidence?

This turns a broad conference idea into an operating question.

The decision also protects the team from attractive distractions. A new dashboard may look useful. A new automation may sound efficient. A new AI workflow may feel current. The better question is whether it improves a decision that matters.

If the answer is yes, continue. If the answer is unclear, pause before building.

One Process

Most Slate pain is process pain wearing a technical costume.

A form is confusing because the process around it is unclear. A report is ignored because the meeting has no use for it. A communication rule fails because no one owns the audience definition. A portal page goes stale because maintenance lacks an owner. A query becomes risky because its assumptions were left undocumented.

Choose one process that matters and map it from beginning to end.

Keep the map simple:

1. What starts the process?
2. What data enters?
3. What rules act on it?
4. What staff touch it?
5. What student experience does it create?
6. What report or review confirms that it worked?
7. What breaks when timing, data, or ownership is unclear?

This is where many teams find the real project hiding underneath the requested project.

The request may be a report. The project may be a meeting routine. The request may be a campaign. The project may be audience governance. The request may be a portal page. The project may be content ownership and review timing.

Map the process before accepting the object as the whole problem.

One Signal

Choose one behavior the office already captures and decide what it should mean.

Event attendance is a signal. No show behavior is a signal. Portal login frequency is a signal. Aid page views are a signal. Inquiry source is a signal. Program interest changes are signals. Reader movement is a signal. Email clicks can be a signal, with care.

The work is to define the signal with restraint.

Ask:

1. What does the signal suggest?
2. What does it fail to prove?
3. Where does it fit in the student journey?
4. Which action should it inform?
5. Who should see it?
6. When should they see it?
7. How should the office review whether the action helped?

A good signal makes one next step smarter rather than trying to explain the whole student.

That restraint matters. A portal visit may suggest interest, confusion, or task completion. Event attendance may suggest commitment, availability, family involvement, or curiosity. A financial aid page view may suggest planning, concern, comparison, or a routine checklist step.

Treat signal as evidence for better judgment, rather than a verdict.

One Campaign

Communications are where many conference ideas become visible. They are also where complexity grows fast.

Choose one campaign and make it cleaner.

Define the audience. Define the purpose. Identify reusable blocks. Name the suppressions. Decide what gets tested. Assign an owner. Decide when it will be reviewed.

That single campaign can become the model for governed personalization.

A team can start with one campaign instead of rebuilding every communication path at once. If the model works, the pattern can spread.

The campaign map should answer:

1. Who belongs in the audience?
2. Who should be suppressed?
3. Which data fields drive personalization?
4. Which blocks should be reusable?
5. Which scenarios should be tested?
6. Who owns content?
7. Who owns audience logic?
8. Who reviews results?

This approach turns personalization into a system the team can understand. It also makes the student experience more coherent because messages are tied to purpose, data quality, and review.

One Hidden Object

Every office has hidden objects that shape daily work.

The rule nobody wants to touch. The lookup value no one remembers creating. The old event template still being copied. The source code that appears in reports and confuses new staff. The dataset with an unclear purpose. The permission group that has grown too broad. The query that runs every week with no visible owner.

Choose one.

Give it a name, an owner, a note, a test, or a retirement plan.

This work rarely gets applause. It saves future time. It reduces future confusion. It makes the instance easier to trust.

A hidden object review should answer:

1. What is this object?
2. Why does it exist?
3. Who owns it?
4. What depends on it?
5. What happens if it changes?
6. What should be tested before change?
7. Should it stay, change, merge, or retire?

Hidden logic deserves visible care.

One Teaching Artifact

The work becomes shared when it can be taught.

A teaching artifact can be simple and still useful. A one page explanation of a campaign rule can be useful. A short walkthrough of how a query was built can be useful. A screenshot with labels can be useful. A checklist for testing a portal update can be useful. A source map can be useful.

The goal is to turn one person’s understanding into team memory.

For the chosen decision, process, signal, campaign, or hidden object, create one teaching artifact that helps someone else understand the work.

It should answer:

1. What is this?
2. Why does it exist?
3. Who uses it?
4. What does it depend on?
5. What can go wrong?
6. What should be tested?
7. Who owns maintenance?

Teaching also reveals whether the build is actually clear. If the owner struggles to explain the work, the system is carrying too much private logic.

That is a risk worth finding early.

One Practitioner Skill

Every project should grow a person or a team.

That is the part offices often skip. The build ships, the next request arrives, and the learning remains informal. The Summit should change that habit.

If the project involves queries, teach row grain. If it involves events, teach signal definition. If it involves communications, teach suppressions and reusable logic. If it involves portals, teach ownership and review. If it involves AI, teach prompt framing, source checking, and human review. If it involves reports, teach decision design.

Choose one skill and name the artifact that will prove growth.

The artifact might be a query pattern, source map, campaign governance map, report note, prompt review checklist, workflow audit, or teach back guide.

Development becomes stronger when it produces evidence.

A Thirty Day Rhythm

The model can run as a thirty day Summit follow up.

Week 1: gather notes, choose the decision, and identify the team members who should be involved.

Week 2: map the process and define the signal.

Week 3: improve one campaign, workflow, report, or hidden object.

Week 4: create the teaching artifact, teach it back, and decide what should become standard practice.

That rhythm keeps the work small enough to complete. It also gives conference learning a real place in office operations.

At the end of the month, the team should have something concrete:

1. A decision statement.
2. A process map.
3. A signal definition.
4. A cleaner campaign or object.
5. A teaching artifact.
6. A practitioner skill target.
7. A decision about what becomes standard.

That is a better use of conference energy than opening a giant idea list and hoping it becomes strategy.

Sort Ideas By Use

Treating every idea the same is how teams get overwhelmed.

Some insights are ready for immediate use. They can become a checklist, naming cleanup, query adjustment, training topic, or meeting prompt.

Some insights need a project plan. They touch multiple teams, data structures, communications, portals, permissions, or reporting.

Some insights should be studied later. They may be promising while local timing, staffing, data quality, or governance lags behind the idea.

Some insights should be shared with someone else. A session may have little value for one team and strong value for another.

A simple sorting frame works:

1. Use now.
2. Plan next.
3. Study later.
4. Share with someone else.

That is enough.

The purpose is momentum with judgment. A team should act on ideas that are ready, plan the ideas that need structure, watch the ideas that need more maturity, and route the ideas that belong elsewhere.

Use The Collection As An Index For Action

This collection should behave like an index for action.

If the team is working on reporting, start with decision infrastructure and the Query Ladder. If the team is working on communications, start with signals and governed personalization. If the team is thinking about events or portals, use the signal system frame. If the instance feels harder to maintain than it should, start with the hidden operating system. If the team is onboarding staff, use the teaching chapter and the practitioner stack. If the team is thinking about AI, read it through governance, source checking, review, and documentation.

That is how the Summit keeps working after the conference ends.

The collection can give different levels of attention to different sessions. A field guide is useful because it helps readers choose paths. Some sessions offer a full operating model. Others offer one naming pattern, one governance question, one query move, one staff exercise, one portal idea, one reporting habit, or one caution that belongs in future work.

That still matters.

The goal is to keep useful ideas findable when a real need appears.

What Leaders Should Do

Leaders should give conference learning an operating path.

Avoid asking staff to return inspired and absorb all translation work privately. Create a short debrief. Ask which decisions need improvement. Choose a small number of ideas. Assign owners. Set review dates. Protect time for documentation and teaching. Make the conference part of the office learning system.

Leaders also need to value the quieter work.

The most important project in a month may be cleaning logic that supports ten other processes. It may be documenting how an audience is built. It may be training staff to understand a report. It may be creating a review habit before personalization grows. It may be naming what a signal means and what it fails to show.

Those moves make future work easier.

What Senior Builders Should Do

Senior builders should bring assumptions home with the idea.

When a session shows a strong build, ask what conditions made it work. What data did it require? What governance did it assume? What staff capacity did it need? What could break? What needed to be taught? What measured success?

That is how local adaptation becomes responsible.

The builder often sees across the system. They see how a form affects a rule, how a rule affects a campaign, how a campaign affects a report, how a report affects a meeting, and how a meeting affects a decision.

That view is powerful. Use it with care.

The best builders translate. They turn feature knowledge into shared understanding. They help leaders see dependencies. They help staff understand the why behind the object. They help the office choose a path that can be maintained.

What Newer Practitioners Should Do

Newer practitioners can start before they understand the whole system.

Start with one rung. Learn how a population is defined. Learn how a rule fires. Learn how a portal page is maintained. Learn how an event record becomes reportable. Learn how a communication audience is built. Learn how to test. Learn how to explain what you built.

That is how expertise grows.

The key is to connect learning to artifacts. Move past reading a help article or watching a session. Build a small object. Explain it. Test it. Ask someone else to use it. Revise it based on what they missed.

Capability grows through practice and review.

AI And Analytics Need The Same Discipline

AI belongs in the work after the Summit because the tools are moving closer to daily practice.

AI can summarize, draft, classify, compare, brainstorm, and explain. It can help create documentation. It can help turn messy notes into a cleaner plan. It can help staff move from blank page to reviewable draft.

It still needs context. It still needs review. It still needs source awareness. It still needs boundaries.

A weak process with AI added to it is still a weak process. A clear process with AI placed carefully inside it can become faster, easier to teach, and easier to scale.

The same is true for analytics.

Better models, richer signals, and smarter dashboards can help enrollment teams see patterns earlier. They can also create false confidence if the office forgets what the data can say and where it reaches its limits. A student is more than a score. A signal is more than a verdict. A model is more than a magic answer.

Good analytics should improve judgment.

The work after the Summit is partly technical and partly ethical. Teams are making choices about how students are understood, grouped, messaged, supported, and measured. Those choices should be visible enough to review.

The Standard

Slate gives offices a lot of power. The Summit shows how creative the community can be with that power. The next step is discipline.

Discipline in naming.

Discipline in testing.

Discipline in ownership.

Discipline in documentation.

Discipline in teaching.

Discipline in review.

That discipline gives creative work somewhere stable to land.

The system is more than what is configured. The system is what people can understand, trust, use, and improve.

That is the standard the work after the Summit should aim for.

Closing

Imagine the first staff meeting after the conference.

Instead of walking in with a long recap, walk in with a small set of choices:

1. Here is one decision we can improve.
2. Here is one process behind that decision.
3. Here is one signal we can define more clearly.
4. Here is one campaign or workflow we can clean up.

5. Here is one hidden object we should document or retire.
6. Here is one thing we can teach the team.
7. Here is one skill we want to build this month.

That meeting would create motion. It would also keep the Summit from becoming a pile of unused notes.

A good Summit sends people home with ideas.

A strong office turns a few of those ideas into habits.

So take the notes. Revisit the sessions. Credit the presenters. Share the examples. Then choose something real.

Choose one decision that needs better support. Choose one process that needs to be mapped. Choose one signal that needs a definition. Choose one campaign that needs governance. Choose one hidden object that needs care. Choose one build that needs to be taught. Choose one skill your team should grow.

That is the work after the Summit.

The Conference Insights Engine ends this opening series with a simple belief: the best Slate work is built, governed, taught, reviewed, and improved in community. Remember to strike true

Find David Dysart and The Innovation Forge on Youtube, Substack, and everywhere podcasts are heard.

You can find the accompanying podcast episodes of The Conference Insights Engine on Slate Summit 2026 at Youtube.