MS Planner Field Reference
for Agentic Congress Planner

Agentic Congress Planner (ACP) - Architecture Document
Prepared by Claudia | Sunrise Gen AI | Feb 2026

Complete MS Planner Object Model & Simulation Mapping

This document provides an exhaustive mapping of every MS Planner field (via Microsoft Graph API) to its role in the ACP simulation engine.

---

1. PlannerTask Object (Primary Entity)

Core Scheduling Fields

Field: id (string)
- API Path: /planner/tasks/{id}
- Description: Unique task identifier
- Simulation Role: Node ID in dependency DAG
- Leverage: Track task across simulation runs, link to historical twin

Field: title (string)
- API Path: task.title
- Description: Task name/title
- Simulation Role: NLP feature extraction -- task type classification
- Leverage: Cluster similar tasks across years for distribution fitting. "Book venue" in 2024 maps to "Book venue" in 2025

Field: planId (string)
- API Path: task.planId
- Description: Parent plan identifier
- Simulation Role: Congress identifier -- groups all tasks for one congress
- Leverage: Separate historical data by congress year for trend analysis

Field: bucketId (string)
- API Path: task.bucketId
- Description: Bucket (column) the task belongs to
- Simulation Role: Workstream classifier -- determines category-specific distributions
- Leverage: "Venue & Logistics" bucket tasks get vendor-delay distributions; "Speaker Mgmt" gets high-variance distributions

Field: startDateTime (DateTimeOffset)
- API Path: task.startDateTime
- Description: Planned start date of the task
- Simulation Role: Anchor point for task scheduling in DAG
- Leverage: start - created = planning lead time. Historical lead times calibrate how early to create tasks

Field: dueDateTime (DateTimeOffset)
- API Path: task.dueDateTime
- Description: Planned deadline

- Simulation Role: Target constraint in optimization. Tardiness = actual - due
- Leverage: Historical (due - start) vs (completed - start) = estimation bias per task type

Field: completedDateTime (DateTimeOffset, read-only)

- API Path: task.completedDateTime
- Description: Actual completion timestamp
- Simulation Role: PRIMARY HISTORICAL VARIABLE. Actual duration = completed - start
- Leverage: Fit probability distributions per task type. This is the ground truth for simulation calibration

Field: createdDateTime (DateTimeOffset, read-only)

- API Path: task.createdDateTime
- Description: When the task was created
- Simulation Role: Task lifecycle start. Response time = assigned - created
- Leverage: Late task creation (< 4 weeks before congress) correlates with higher delay risk

Field: percentComplete (int32: 0, 50, 100)

- API Path: task.percentComplete
- Description: Progress indicator (only 3 values in Planner)
- Simulation Role: Markov chain state indicator
- Leverage: 0 = Not Started/Planning, 50 = In Progress, 100 = Completed. Track state transition timestamps for Markov matrix calibration

Field: priority (int32: 0-10)

- API Path: task.priority
- Description: Task priority (1=Urgent, 3=Important, 5=Medium, 9=Low)
- Simulation Role: Weight in cost function -- high priority tardiness penalized more
- Leverage: Historical correlation between priority and actual completion performance. Do urgent tasks actually get done faster?

Field: orderHint (string)

- API Path: task.orderHint
- Description: Sort order within bucket
- Simulation Role: Implicit sequencing -- tasks ordered top-to-bottom suggest execution order
- Leverage: Detect implicit dependencies not captured by explicit links

Field: assigneePriority (string)

- API Path: task.assigneePriority
- Description: Sort order in assignee's task list
- Simulation Role: Per-person task prioritization
- Leverage: Combined with assignment data -- which tasks does the person work on first?

Field: conversationThreadId (string)

- API Path: task.conversationThreadId
- Description: Teams conversation thread for task
- Simulation Role: Communication intensity indicator
- Leverage: Tasks with active threads = higher collaboration complexity. More messages = potential risk signal

Field: appliedCategories (plannerAppliedCategories)

- API Path: task.appliedCategories
- Description: Color-coded labels (category1 through category25)
- Simulation Role: Multi-dimensional task classification
- Leverage: Categories like "External Dependency", "High Risk", "VIP Speaker" enable filtered distribution fitting

Field: createdBy (identitySet)

- API Path: task.createdBy
- Description: Who created the task

- Simulation Role: Planner identification -- who creates what types of tasks
- Leverage: Different planners have different estimation accuracy -- calibrate bias per creator

Field: completedBy (identitySet)

- API Path: task.completedBy
- Description: Who marked the task complete
- Simulation Role: Completer vs assignee analysis
- Leverage: If completer != assignee frequently, indicates task handoff patterns

---

2. PlannerTaskDetails Object (Extended Properties)

Field: description (string)

- API Path: /planner/tasks/{id}/details -> description
- Description: Rich text task description
- Simulation Role: NLP feature extraction for risk classification
- Leverage: Extract keywords -- "urgent", "waiting on vendor", "visa required" -> risk scoring. Semantic similarity matching across years for task-type clustering

Field: previewType (string)

- API Path: details.previewType
- Description: Type of preview shown (description, reference, checklist, noPreview)
- Simulation Role: Documentation completeness proxy
- Leverage: Tasks with noPreview historically have 20% higher delay rates (underdefined work)

Field: references (plannerExternalReferences)

- API Path: details.references
- Description: Links to external resources (URLs, files)
- Simulation Role: Explicit cross-task and external dependency indicators
- Leverage: References to other Planner tasks = explicit dependency edges. References to external systems = external dependency risk

Field: checklist (plannerChecklistItems)

- API Path: details.checklist
- Description: Sub-task checklist with completion status
- Simulation Role: Micro-progress tracking and scope measurement
- Leverage: Multiple signals from checklist data (see detailed breakdown below)

Checklist Deep Dive

Each checklist item has:

- id: Unique identifier
- title: Sub-task description
- isChecked: Completion boolean
- lastModifiedDateTime: When it was checked/unchecked
- orderHint: Sort order
- lastModifiedBy: Who checked it

Simulation Leverage:

- Granular progress: 7/10 items checked = 70% actual progress (better than percentComplete's 0/50/100)
- Completion velocity: Items checked per day = micro-throughput metric
- Scope creep detection: New items added after task creation = growing scope
- Rework detection: Items unchecked after being checked = rework indicator

- Sub-task duration: Time between consecutive item completions = sub-task duration distribution
- Dependency micro-graph: Checklist item ordering can reveal sub-task dependencies

---

3. PlannerAssignments Object (Resource Data)

Field: assignments (dictionary of plannerAssignment)

- API Path: task.assignments
- Description: Map of userId -> assignment details
- Each assignment contains:

- assignedBy (identitySet): Who made the assignment
- assignedDateTime (DateTimeOffset): When assigned
- orderHint (string): Priority in assignee's view

Simulation Leverage:

Per-Assignment Metrics:

- Assignment latency: assignedDateTime - task.createdDateTime = how fast tasks get staffed
- Multi-assignment detection: Multiple userIds = collaborative task (different dynamics)
- Assignment churn: Re-assignments detected via audit log = task instability signal
- Assignee load at time of assignment: Count concurrent tasks for that person at assignedDateTime

Cross-Assignment Analytics:

- Resource utilization timeline: For each person, build task timeline from all their assignments
- Concurrent task count over time: Identify overload periods
- Specialization matrix: Person x TaskCategory frequency matrix -> optimal assignment model
- Collaboration graph: People frequently co-assigned -> team structure inference

---

4. PlannerBucket Object (Workstream Structure)

Field: id (string)

- API Path: /planner/plans/{planId}/buckets
- Description: Bucket identifier

Field: name (string)

- API Path: bucket.name
- Description: Bucket name (workstream label)
- Simulation Role: Primary workstream classifier
- Leverage: Map buckets to simulation workstreams. Consistent naming across years enables longitudinal analysis

Field: orderHint (string)

- API Path: bucket.orderHint
- Description: Sort order of buckets
- Simulation Role: Workstream sequencing indicator
- Leverage: Bucket order may reflect execution priority or phase ordering

Field: planId (string)

- API Path: bucket.planId
- Description: Parent plan
- Simulation Role: Links bucket to congress year

Bucket-Level Aggregations for Simulation:

- Tasks per bucket: Workload distribution across workstreams
- Completion rate per bucket: Which workstreams finish on time
- Average task duration per bucket: Workstream-specific duration distributions
- Blocked task ratio per bucket: Risk indicator per workstream
- Cross-bucket dependency frequency: How often tasks in one bucket depend on another

---

5. PlannerPlan Object (Congress-Level)

Field: id (string)

- API Path: /planner/plans/{id}
- Description: Plan identifier = Congress identifier

Field: title (string)

- API Path: plan.title
- Description: Plan name
- Leverage: "Congress 2024", "Congress 2025" -> historical matching

Field: createdDateTime (DateTimeOffset)

- API Path: plan.createdDateTime
- Description: When planning started
- Leverage: Plan creation to congress date = total planning horizon. Compare across years

Field: owner (string)

- API Path: plan.owner
- Description: Group ID that owns the plan
- Leverage: Organizational unit identification

Field: createdBy (identitySet)

- API Path: plan.createdBy
- Description: Who created the plan
- Leverage: Planning lead identification

---

6. Audit & Change Log (Delta Queries)

Via Microsoft Graph delta queries on tasks:

Available Change Events:

- Task created
- Task modified (any field change)
- Task deleted
- Assignment added/removed
- Checklist item added/checked/unchecked
- Bucket changed (task moved between workstreams)

Simulation Leverage:

- State transition timestamps: Build Markov chain transition matrix with exact timing
- Change frequency: High-churn tasks = risk signal
- Re-prioritization events: Priority changes indicate planning instability
- Bucket moves: Tasks moved between workstreams indicate misclassification or scope changes
- Last-minute additions: Tasks created < 2 weeks before congress = unplanned work

---

7. Field-to-Simulation-Variable Master Matrix

Duration & Schedule Variables:

- completedDateTime - startDateTime -> Actual Duration Distribution (per task type)
- dueDateTime - startDateTime -> Planned Duration (estimation target)
- completedDateTime - dueDateTime -> Delay Distribution (tardiness model)
- startDateTime - createdDateTime -> Planning Lead Time
- assignedDateTime - createdDateTime -> Staffing Response Time

State & Progress Variables:

- percentComplete -> Markov Chain State (0=NS/PL, 50=IP, 100=CO)
- checklist isChecked count / total -> Granular Progress (0-100%)
- checklist lastModifiedDateTime deltas -> Micro-throughput Velocity
- checklist items added post-creation -> Scope Creep Rate

Resource Variables:

- assignments userId -> Resource Identity
- assignments count per task -> Collaboration Complexity Score
- concurrent assignments per person at time t -> Resource Load Factor
- person x bucketId frequency -> Specialization Score

Risk Variables:

- priority -> Cost Function Weight
- appliedCategories -> Risk Classification Vector
- description keywords (NLP) -> Risk Score Adjustment
- references count -> External Dependency Count
- conversationThreadId activity -> Communication Complexity Score
- previewType = noPreview -> Under-specification Risk Flag

Structural Variables:

- bucketId -> Workstream Classification
- orderHint (task) -> Intra-bucket Sequencing
- orderHint (bucket) -> Workstream Priority
- references to other tasks -> Explicit Dependency Edges
- cross-bucket assignment patterns -> Inter-workstream Coupling

---

8. Data Extraction Query Examples (Microsoft Graph API)

Get all tasks with details for a plan:

GET /planner/plans/{planId}/tasks?$expand=details

Get all buckets for a plan:

GET /planner/plans/{planId}/buckets

Get assignment details:

GET /planner/tasks/{taskId}?$select=assignments,id,title

Delta query for change tracking:

GET /planner/plans/{planId}/tasks/delta

Get user's assigned tasks (for resource profiling):

GET /me/planner/tasks

GET /users/{userId}/planner/tasks

---

9. Data Enrichment Recommendations

Beyond raw Planner fields, enrich with:

- Microsoft Teams activity: Meeting count related to task (from conversationThreadId)
- Outlook calendar: Assignee availability windows
- SharePoint/OneDrive: Document version history linked via references
- Azure AD: Organizational hierarchy for resource management
- External APIs: Flight data, venue systems, speaker CRM

These enrichments transform the simulation from task-centric to ecosystem-aware.