**Пояснения к блокам:**

1. Событие: Создание/Обновление тикета

* Триггеры: создание тикета, изменение очереди или статуса.
* Примеры событий: TicketCreate, TicketQueueUpdate, TicketStateUpdate.

1. Проверка участия очереди

* Использует метод TicketAllocationList().
* Проверяет, подключена ли очередь к автоматическому распределению через настройки.

1. Конфигурационные проверки

* Проверяет параметры в Kernel/Config.pm:
* NotAutoAssignWhenNewOwnerIsSelected — ручной выбор владельца.
* ReAssign — перераспределение при изменении статуса.

1. Проверка статуса тикета

* Сверяет статус с разрешенными в настройках очереди (StateIDs).
* Пример: распределение только для статусов "открыт" и "в работе".

1. Поиск исполнителя

* Алгоритм метода TicketAllocationUserGet():
* Фильтрация пользователей:
* Исключение ответственных из настроек (ResponsibleIDs).
* Проверка нахождения на рабочем месте (OutOfOffice).
* Учет исключенных пользователей (ExcludedUserIDs).
* Расчет компетенций (CompetenceLevel).
* Учет текущей загрузки (Limit — максимальное кол-во тикетов).

1. Очередь ожидания

* Если исполнитель не найден, тикет добавляется в таблицу ticket\_allocation\_queue.
* Параметр WaitPeriod определяет интервал повторной проверки.

1. Периодическая проверка (AllocationCheck)

* Запускается через cron задачу.
* Проверяет тикеты в очереди с истекшим WaitPeriod.
* Использует тот же алгоритм поиска исполнителя.

**Краткий абстрактный пример работы автоматического распределения:**

1. Пользователь создает тикет в очереди "Поддержка".
2. Система проверяет, что очередь "Поддержка" подключена к автораспределению.
3. Проверяется, не был ли владелец выбран вручную.
4. Определяется, что статус "новый" разрешен для распределения.
5. Система ищет свободных агентов с учетом их загрузки (не более 10 тикетов).
6. Если агент найден — тикет назначается, иначе добавляется в очередь ожидания.
7. Через 15 минут cron задача AllocationCheck повторно пытается распределить тикет.