ЛЕКЦИЯ № 12 · Проект MATLAB IoT

ThingSpeak как интерфейс MATLAB-аналитики и M2M-взаимодействия



Бельгибаев Бауржан Абдрахимович
Д.т.н., доцент
Кафедра «Искусственный интеллект и big data»
ОП 6B07113 «Интеллектуальные системы управления»

Цель лекции

Изучить роль ThingSpeak как облачного интерфейса обмена данными между устройствами и средой MATLAB, рассмотреть модели M2M-коммуникации, способы организации каналов, автоматизированного анализа временных рядов и передачи результатов. Сформировать умение проектировать воспроизводимый и безопасный учебный M2M-сценарий.

План лекции

1. Сущность M2M и место ThingSpeak
2. Архитектура устройств, каналов и MATLAB
3. Структура сообщений и модель данных
4. REST API, запись и чтение данных
5. MATLAB Analysis и MATLAB Visualizations
6. Автоматизация по расписанию и событиям
7. Алгоритмы обработки измерений
8. Передача аналитических результатов между устройствами
9. Задержки, периодичность, целостность и QoS
10. Безопасность и обработка отказов
11. Проектирование M2M-сценария
12. Сквозной пример: контроль температуры двух узлов

1 Сущность M2M и место ThingSpeak

M2M (machine-to-machine) означает обмен информацией между техническими устройствами и программными компонентами без обязательного участия человека в каждой операции. Например, измерительный узел передаёт показания серверу, а аналитический компонент рассчитывает диагностический показатель и делает его доступным другому узлу.

M2M является частью более широкой концепции IoT, которая включает сетевые устройства, облачную инфраструктуру, управление доступом, хранение данных и пользовательские приложения. При этом M2M не обязательно использует интернет: взаимодействие возможно и в локальных промышленных сетях.

ThingSpeak предоставляет облачные каналы для хранения временных рядов и средства интеграции с MATLAB. В учебной системе он может играть роль посредника между источником данных и аналитическим потребителем. Однако ThingSpeak не следует считать заменой промышленному контроллеру или системе аварийной защиты.

1 Сущность M2M и место ThingSpeak (продолжение)

Практический пример: датчик температуры на Raspberry Pi публикует измерения в канал A; MATLAB анализирует ряд и публикует вычисленный показатель в канал B; второй программный узел считывает канал B для отображения состояния. Это демонстрация информационного M2M-взаимодействия, а не гарантированного управления в реальном времени.

Основные преимущества схемы — разделение устройств и аналитики, доступ к истории и воспроизводимость вычислений. Ограничения — зависимость от сети, задержки, квоты сервиса и необходимость защиты ключей.

2 Архитектура устройств, каналов и MATLAB

Логическая структура состоит из источника измерений, транспортного уровня, облачного хранилища, аналитического блока и потребителя результата. Источником может быть Arduino с сетевым модулем, Raspberry Pi или программный симулятор; потребителем — панель мониторинга, другое устройство или программа MATLAB.

Канал ThingSpeak содержит записи с временными метками и значениями полей. Для учебной задачи удобно выделить Field 1 для температуры, Field 2 для влажности, Field 3 для состояния. Если аналитический результат должен быть доступен другим компонентам, его можно сохранять в отдельном канале с независимыми правами доступа.

Архитектура «измерение → облако → анализ → результат» не гарантирует мгновенную реакцию. Для аварийных блокировок используется локальный контур управления; облачный контур предназначен прежде всего для наблюдения, диагностики и некритичных рекомендаций.

При проектировании задают частоту измерения, допустимую задержку, структуру сообщений, правила проверки данных, способ хранения ключей и действия при отсутствии сети. Без этих решений даже работающий демонстрационный прототип трудно масштабировать.

2 Архитектура устройств, каналов и MATLAB (продолжение)

Поток данных удобно документировать таблицей: источник, название поля, единица, диапазон, период обновления, получатель, правило обработки и статус доступа.

3 Структура сообщений и модель данных

Содержимое M2M-сообщения должно быть однозначным. Для температуры необходимо указать значение, единицу измерения, время измерения и идентификатор источника. При передаче через канал часть метаданных хранится в настройках, но формат должен быть описан в проектной документации.

JSON широко используется в веб-API благодаря именованным полям. Например, объект может содержать device_id, timestamp_utc, temperature_C, humidity_pct и sequence. Последовательный номер помогает выявить пропуски и дубли при повторной отправке.

Нужно различать время измерения и время записи на сервере. При сетевой задержке эти моменты не совпадают. Для расчёта скорости изменения параметра предпочтительна исходная временная метка датчика, если она достоверна.

Некорректные данные (NaN, Inf, значения вне физического диапазона) не должны без проверки использоваться в вычислениях. Если поле отсутствует, следует сохранить признак пропуска, а не заменять его нулём.

3

Структура сообщений и модель данных (продолжение)

Единицы измерения особенно важны при объединении источников: температура в °C и K, давление в Па и кПа требуют явного преобразования. Формально одинаковые числа без единиц не гарантируют сопоставимость.

```
% MATLAB: структура M2M-сообщения
msg = struct('device_id','node_A', ...
    'timestamp_utc','2026-11-09T08:00:00Z', ...
    'temperature_C',26.4, ...
    'humidity_pct',61.2,'sequence',101);
encoded = jsonencode(msg);
disp(encoded);
decoded = jsondecode(encoded);
assert(isfinite(decoded.temperature_C));
```

4 REST API: запись и чтение данных

ThingSpeak предоставляет HTTP-интерфейсы для записи и чтения данных каналов. При записи используются идентификатор канала, ключ записи и значения полей; при чтении закрытого канала необходим соответствующий ключ чтения. Конкретные URL и допустимые параметры следует сверять с текущей документацией.

При отправке запросов устройство должно анализировать код HTTP-ответа и содержимое ответа сервиса. Успешное соединение с сервером не означает, что запись действительно сохранена. Следует также учитывать ограничения частоты обновления, зависящие от условий использования.

В MATLAB доступна функция thingSpeakRead для получения данных и thingSpeakWrite для публикации при наличии соответствующих возможностей среды и прав доступа. Для лабораторного занятия лучше сначала выполнить чтение общедоступного учебного канала или работать с синтетическим массивом.

В программном коде нельзя оставлять реальные ключи доступа. Их получают из защищённой конфигурации. Для закрытого канала следует использовать только выданные владельцем ключи.

4 REST API: запись и чтение данных (продолжение)

Сетевые ошибки требуют ограниченных повторных попыток и локальной буферизации. Повторная передача должна учитывать возможные дубли, а при загрузке исторических данных — поддержку исходных временных меток конкретным API.

```
% MATLAB: шаблон чтения канала (нужен реальный ID)
channelID = 123456;  % заменить
readKey = 'YOUR_READ_API_KEY';
% При наличии доступа и данных:
% [values,times] = thingSpeakRead(channelID, ...
%     'Fields',1,'NumPoints',100,'ReadKey',readKey);
% Автономная демонстрация без обращения к сервису:
t = (0:20:140)';
T = [24.2;24.6;25.0;25.5;26.1;26.7;27.2;27.8];
plot(t,T,'-o'); grid on;
xlabel('Время, с'); ylabel('Температура, °C');
```

5 MATLAB Analysis и MATLAB Visualizations

MATLAB Analysis позволяет выполнять вычисления над данными ThingSpeak в доступной конфигурации сервиса. MATLAB Visualizations предназначен для создания пользовательских графиков на основе данных канала. Конкретные способы запуска и публикации результатов зависят от функций аккаунта.

Типовой аналитический сценарий состоит из чтения последних N измерений, проверки качества, вычисления среднего и максимума, формирования логического признака превышения и отображения графика. Для научного отчёта нужно сохранить используемые пороги и параметры обработки.

Формула среднего значения: $\bar{T} = (1/n)\Sigma T_i$. Выборочное стандартное отклонение: $s = \sqrt{[\Sigma(T_i - \bar{T})^2/(n-1)]}$. Для оценки изменений за интервал используют $v_i = (T_i - T_{i-1})/(t_i - t_{i-1})$, где t измеряется в секундах.

Если ряд содержит пропуски, анализ должен явно определять политику обработки: исключение, интерполяция или отдельный сигнал о недостоверности. Интерполяция допустима не во всех инженерных задачах, поскольку может скрыть кратковременное опасное событие.

5

MATLAB Analysis и MATLAB Visualizations (продолжение)

Скользящее среднее movmean уменьшает случайные колебания, но вводит сглаживание и потенциальную задержку. Поэтому для аварийного контроля нельзя полагаться исключительно на сглаженный облачный показатель.

```
% MATLAB: обработка полученного ряда
T = [24.8 25.1 NaN 26.2 27.4 28.3 29.1];
valid = isfinite(T);
Tv = T(valid);
meanT = mean(Tv);
maxT = max(Tv);
countHigh = sum(Tv >= 28);
fprintf('Средняя %.2f, максимум %.2f, превышений %d\n', ...
    meanT,maxT,countHigh);
```

6 Автоматизация аналитики: расписание и события

Анализ облачных данных может запускаться вручную, по расписанию либо в ответ на событие, если соответствующие возможности доступны в используемой версии ThingSpeak. Сценарии расписания подходят для периодического формирования сводок, а событийные механизмы — для проверки определённых условий.

При выборе периодичности необходимо учитывать скорость изменения технологического процесса и ограничения сервиса. Например, анализ температуры помещения раз в несколько минут может быть достаточен для учебного мониторинга, но не для быстрого аварийного отключения нагревателя.

Событийный сценарий следует проектировать с учётом повторов. Если одно и то же условие сохраняется длительное время, программа может многократно выдавать одинаковое сообщение. Для уменьшения повторов используют фиксацию состояния, гистерезис или ограничение частоты уведомлений.

Автоматическое выполнение MATLAB-кода не означает, что устройство получило команду. Результат анализа нужно явно записать в доступный канал или передать через предусмотренный API, после чего принимающая сторона должна его прочитать и проверить.

6 Автоматизация аналитики: расписание и события (продолжение)

Для воспроизводимости следует сохранять версию алгоритма, период запуска, идентификатор входного канала и правила формирования результата. Это позволяет объяснить, почему одинаковые данные могли привести к разным выводам после изменения настроек.

7 Алгоритмы обработки измерений и принятия решений

В M2M-системах решения часто строятся на пороговых правилах, статистических оценках и сравнении нескольких датчиков. Простейшее правило: alarm = (T ≥ Tmax). Если температура достигает порога, формируется предупреждение. Значение порога должно быть обосновано условиями объекта.

Для уменьшения частых переключений при шуме используется гистерезис. При верхнем пороге Ton предупреждение включается, а при нижнем Toff < Ton — снимается. Между порогами сохраняется прежнее состояние. Такой алгоритм имеет память.

Для проверки нескольких параметров применяются логические операции AND и OR. Например, warning = (T > 30) || (H > 75). Перед сравнением следует проверить isfinite(T) и isfinite(H), иначе NaN может ошибочно восприниматься как отсутствие превышения.

Сравнение двух узлов позволяет выявлять пространственные различия. Для температуры $\Delta T = TA - TB$. Однако большое ΔT не всегда означает неисправность: датчики могут находиться в разных условиях. Необходимы сведения о размещении и точности.

При оценке качества прогноза можно использовать $MAE=(1/n)\Sigma|y_i-\hat{y}_i|$ и $RMSE=\sqrt{[(1/n)\Sigma(y_i-\hat{y}_i)^2]}$. Эти метрики требуют эталонных или проверочных значений, а не только одного временного ряда.

7

Алгоритмы обработки измерений и принятия решений (продолжение)

```
% MATLAB: гистерезис по температуре
T = [27.5 28.1 28.5 27.9 27.6 26.8 28.2];
Ton = 28; Toff = 27;
alarm = false; states = false(size(T));
for k = 1:numel(T)
    if ~alarm && T(k)>=Ton
        alarm = true;
    elseif alarm && T(k)<=Toff
        alarm = false;
    end
    states(k) = alarm;
end
disp(table(T(:),states(:), ...
    'VariableNames',{'Temperature_C','Alarm'}));
```

8

Передача аналитических результатов между устройствами

После расчёта MATLAB может сформировать информационное сообщение: средняя температура, тренд, признак предупреждения или диагностический индекс. Для передачи другому компоненту результат публикуется в выделенном канале или через предусмотренный интерфейс.

Принимающий узел должен проверять не только значение результата, но и его актуальность. Например, сообщение «норма», рассчитанное час назад, не должно использоваться как подтверждение текущего состояния. Для этого применяются временные метки и предельный возраст записи.

Необходимо различать рекомендацию и управляющую команду. Облачный признак warning может служить основанием для уведомления оператора, но непосредственное включение или отключение опасного оборудования требует локальных проверок и защит.

Для устойчивого M2M-обмена полезны идентификатор источника, номер сообщения, временная метка, версия формата и код состояния. При изменении формата следует обеспечивать совместимость отправителя и получателя.

При передаче результатов между каналами важно соблюдать права доступа. Чтение публичного канала не должно автоматически предоставлять возможность записи в канал с управляющими параметрами.

9 Задержки, периодичность и целостность данных

Полная задержка облачной аналитики складывается из времени измерения, ожидания передачи, сетевой задержки, записи в сервис, ожидания запуска анализа и доставки результата. Условно L = Lmeasure + Lnetwork + Lstore + Lanalysis + Lreturn. Для конкретного проекта эти составляющие нужно измерять экспериментально.

Период опроса датчика и период обновления канала могут различаться. Если измерения формируются чаще, чем допускает облачная публикация, требуется буферизация или агрегирование. При агрегировании часть исходной информации теряется, что следует указать в отчёте.

Целостность ряда проверяют по числу ожидаемых и полученных записей. Доля потерь P_loss = (Nexpected−Nreceived)/Nexpected × 100 %, если ожидаемое число корректно определено и дубли предварительно исключены.

Для временных рядов полезно анализировать интервалы между метками $\Delta t_i = t_i - t_{i-1}$. Большие интервалы могут означать потерю сообщений, отключение устройства либо штатное изменение режима измерения. Без журналов нельзя однозначно установить причину.

9 Задержки, периодичность и целостность данных (продолжение)

ThingSpeak удобен для учебных и многих мониторинговых задач, однако не следует приписывать ему гарантированное время доставки или жёсткие свойства реального времени без подтверждённой спецификации и испытаний.

10

Безопасность, конфиденциальность и отказоустойчивость

Для работы с закрытыми каналами используются ключи чтения и записи. Ключи необходимо хранить в защищённой конфигурации, не публиковать в исходном коде и ограничивать круг пользователей, имеющих доступ. При подозрении на раскрытие ключ следует заменить.

Передачу рекомендуется выполнять по HTTPS. Raspberry Pi должен регулярно обновляться, использовать надёжную аутентификацию и не предоставлять ненужные сетевые службы. Ошибки авторизации следует отличать от временных сетевых отказов.

При потере сети источник продолжает локальное измерение и сохраняет данные в журнале. После восстановления соединения возможно восстановление истории, если выбранный API и ограничения аккаунта это допускают. Повторная отправка должна предотвращать незаметное дублирование.

Если данные недостоверны, аналитический модуль формирует отдельный статус «ошибка измерения» вместо состояния «норма». Это особенно важно при NaN, пропущенных значениях и несогласованных единицах.

Физическая безопасность обеспечивается локальным контуром. Облачный M2M-обмен не должен быть единственным механизмом отключения нагревателя, двигателя или другого опасного оборудования.

11 Методика проектирования учебного M2M-сценария

Этап 1. Определить объект и измеряемые параметры. Для микроклимата выбирают температуру и влажность, задают единицы и допустимые диапазоны. Этап 2. Составить структурную схему: узел A → ThingSpeak → MATLAB → канал результата → узел B.

Этап 3. Создать каналы с понятными названиями полей. Разделить ключи записи и чтения, зафиксировать идентификаторы каналов в локальной документации, не раскрывая секретные значения. Этап 4. Проверить передачу искусственных данных.

Этап 5. Реализовать проверку isfinite, вычисление среднего, максимума и признака превышения. Этап 6. Добавить визуализацию и таблицу результатов. Этап 7. Проверить, как система реагирует на пропуски и потерю сети.

Этап 8. Заменить искусственные данные реальными измерениями, документировать модель датчика и схему подключения. Этап 9. Провести повторные испытания и сравнить полученные временные ряды с исходным локальным журналом.

Критерии оценки: корректность структуры сообщений, воспроизводимость расчётов, понятность графиков, обнаружение ошибок, соблюдение ограничений платформы и защита API-ключей.

12

Сквозной пример: два узла мониторинга температуры

Рассмотрим два условных измерительных узла A и B, размещённых в разных зонах помещения. Каждый узел передаёт температуру в собственный канал. MATLAB получает ряды, проверяет данные и рассчитывает разность температур. При $|T_A - T_B| \geq 2$ °C формируется учебный диагностический признак.

Для простоты предполагаем, что временные метки синхронизированы и обе серии содержат измерения в одинаковые моменты. В реальном проекте требуется сопоставить данные по времени, например с помощью timetable и retime/synchronize, а также выбрать допустимое расхождение временных меток.

Порог 2 °C является учебным и не означает неисправность. Для инженерного вывода необходимо учитывать точность датчиков, тепловую неоднородность помещения и режим вентиляции.

Результат анализа может быть опубликован в отдельном канале, но в автономном примере ниже публикация не выполняется. Это позволяет проверить вычислительную часть без сетевого доступа и API-ключей.

В отчёте следует представить графики двух температур, график разности, количество диагностических событий и обсуждение ограничений выбранного критерия.

12

Сквозной пример: два узла мониторинга температуры (продолжение)

```
% MATLAB: воспроизводимый пример двух узлов
clear; clc; close all;
t = (0:20:140)';
TA = [24.1;24.5;25.0;25.7;26.1;27.0;27.5;28.1];
TB = [23.9;24.0;24.2;24.4;24.5;24.7;24.9;25.1];
assert(numel(TA)==numel(TB) && numel(TA)==numel(t));
valid = isfinite(TA) & isfinite(TB);
delta = TA-TB;
limit = 2.0;
flag = valid & abs(delta)>=limit;
fprintf('Диагностических событий: %d\n',sum(flag));
result = table(t,TA,TB,delta,flag);
disp(result);
figure; plot(t,TA,'-o',t,TB,'-s','LineWidth',1.4);
grid on; xlabel('Время, с'); ylabel('Температура, °C');
legend('Узел A','Узел B','Location','best');
title('Сравнение двух IoT-узлов');
```

12

Сквозной пример: два узла мониторинга температуры (продолжение)

```
figure; plot(t,delta,'-o'); hold on;
yline(limit,'--','Порог +2 °C');
yline(-limit,'--','Порог -2 °C');
grid on; xlabel('Время, с');
ylabel('Разность температур, °C');
```

3 Заключение

ThingSpeak позволяет организовать обмен измерениями между IoT-устройствами и MATLAB-аналитикой через облачные каналы. M2M-сценарий требует не только передачи чисел, но и согласованной структуры данных, временных меток, проверки достоверности, разграничения доступа и обработки отказов. MATLAB предоставляет средства статистического анализа, визуализации и формирования диагностических признаков. При этом облачный обмен не заменяет локальные контуры безопасного управления.

Контрольные вопросы (1/2)

1. Что означает M2M и как оно связано с IoT?
2. Какие функции ThingSpeak выполняет в M2M-архитектуре?
3. Чем отличается измерительный узел от аналитического?
4. Что такое канал ThingSpeak?
5. Какие метаданные необходимы для интерпретации измерения?
6. Почему следует указывать единицы измерения?
7. Для чего используется порядковый номер сообщения?
8. Чем отличается время измерения от времени записи в облако?
9. Какие функции выполняют REST API?
10. Зачем нужны ключи чтения и записи?
11. Какие задачи решает MATLAB Analysis?
12. Для чего предназначен MATLAB Visualizations?
13. Как рассчитывается среднее значение ряда?
14. Как обработать NaN и Inf?
15. Каковы достоинства и недостатки скользящего среднего?

Контрольные вопросы (2/2)

16. Чем отличаются запуск по расписанию и событийный запуск?

17. Что такое гистерезис?

18. Как аналитический результат передаётся другому узлу?

19. Почему важно проверять актуальность сообщения?

20. Из каких составляющих складывается задержка M2M?

21. Как оценить долю потерь записей?

22. Для чего нужна локальная буферизация?

23. Какие риски связаны с публикацией API-ключей?

24. Почему облачный канал не подходит как единственная аварийная защита?

25. Какие материалы входят в отчёт по M2M-проекту?

Д Задания для самостоятельной работы

1. Постройте структурную схему M2M-системы с двумя датчиками, ThingSpeak и MATLAB; подпишите направления потоков.

2. Сформируйте 20 JSON-сообщений с идентификатором устройства, временем, температурой и порядковым номером.

3. Разработайте MATLAB-скрипт, рассчитывающий среднее, максимум и число превышений для учебного ряда с пропусками.

4. Реализуйте гистерезис для двух порогов температуры и сравните число переключений с простым пороговым алгоритмом.

5. Предложите правила буферизации и повторной отправки при отключении сети на 10 минут.

6. Для двух температурных каналов рассчитайте разность, выявите интервалы расхождения и подготовьте инженерное объяснение.

Рекомендуемые источники

- Силлабус дисциплины «Проект MATLAB IoT», лекция №12.
- MathWorks. ThingSpeak Documentation — https://www.mathworks.com/help/thingspeak/
- MathWorks. ThingSpeak REST API — https://www.mathworks.com/help/thingspeak/rest-api.html
- MathWorks. MATLAB Documentation — https://www.mathworks.com/help/matlab/
- Raspberry Pi Documentation — https://www.raspberrypi.com/documentation/