IMS Europe LTI Bootcamp

Martin Lenord – Turnitin

Bracken Mosbacker – IMS Global

Coming Up

Intros and goals for the session

High level look at LTI Advantage

Demos and where to find resources

30 minute break to get coffee

Deep dive into building an LTI Advantage app

Intros


Quick Fire Intros
Name & Company
Tool or Platform
Programming Language
Current LTI Knowledge

Coming Up

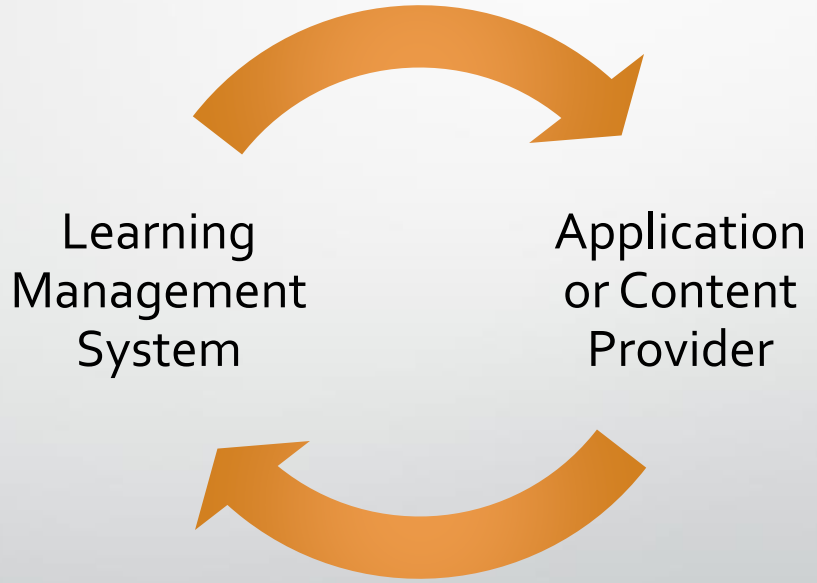
What is LTI and what does it aim to solve?

Why use LTI 1.3?

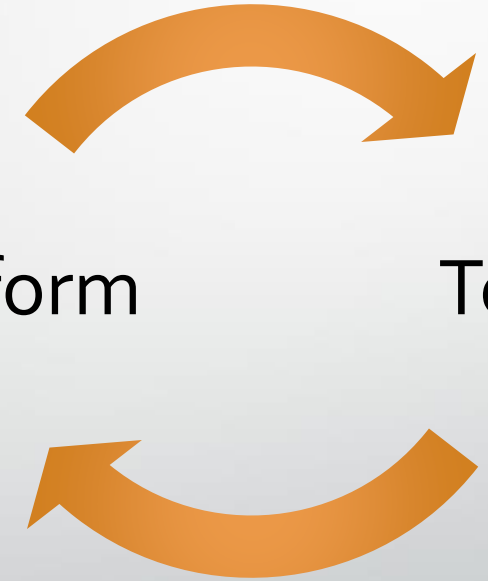
What is LTI Advantage?

What is next for LTI?

What Does LTI Do?



What Does LTI Do?



Platform Tool

What Types of Content?

Assignment Workflows

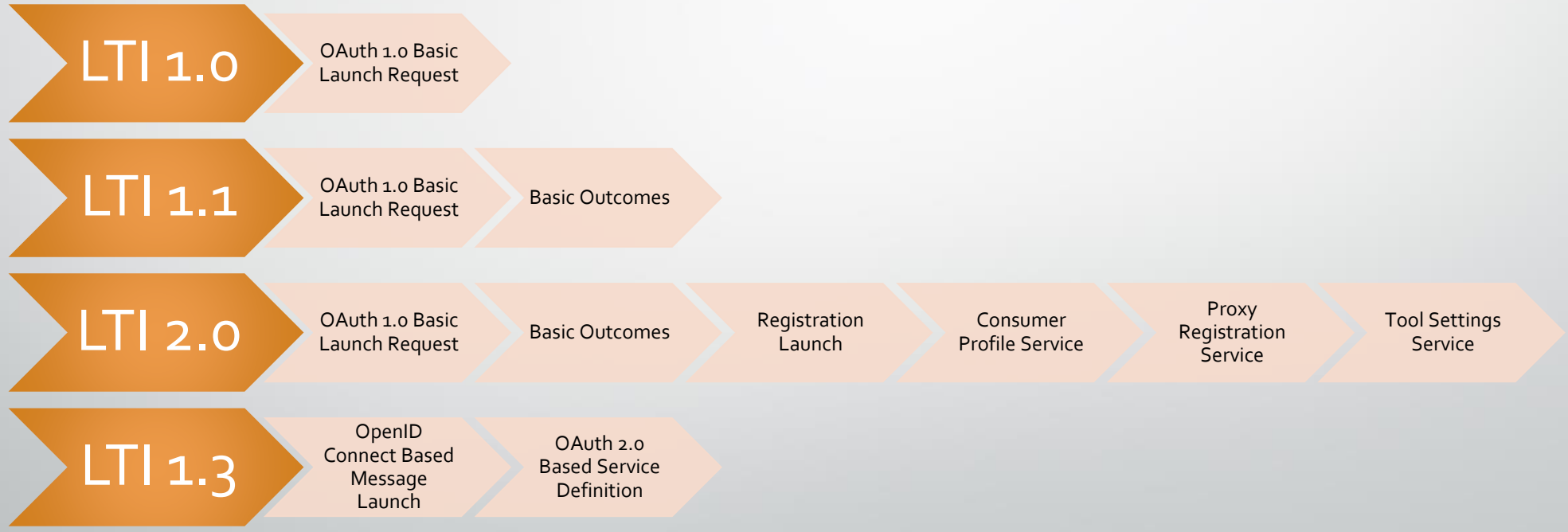
Publisher Content

Games

Utilities

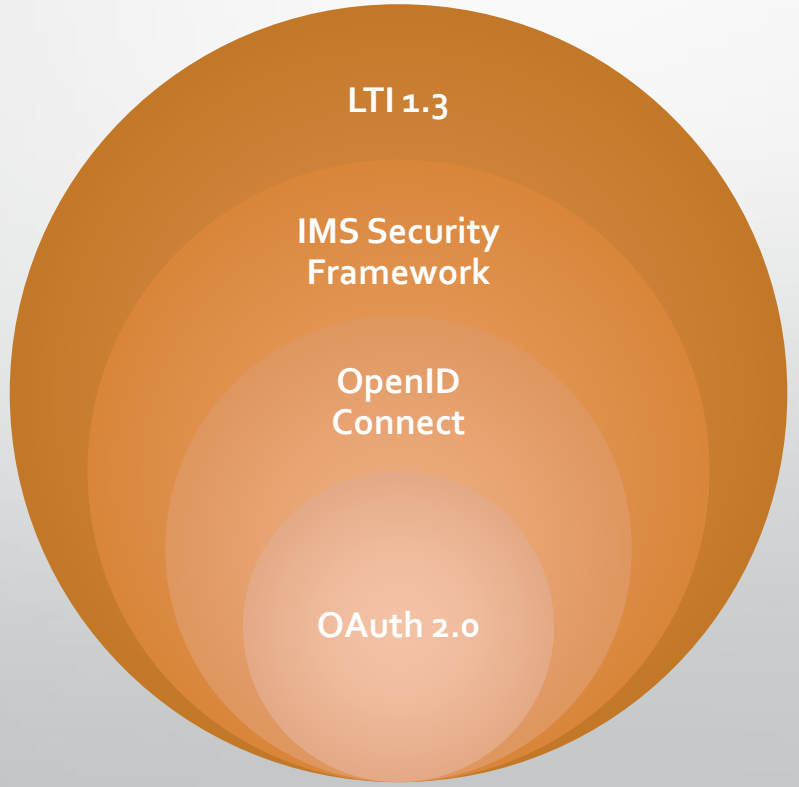
LTI 1.3 and LTI Advantage

Why Use LTI 1.3?



Why Use LTI 1.3?

Security



Why Use LTI 1.3?

Extensibility

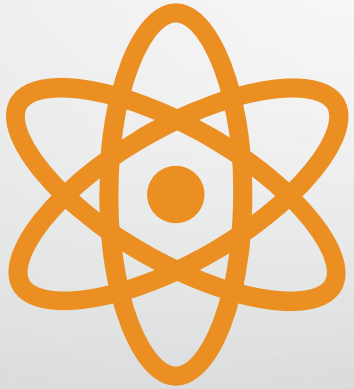
LTI 1.3 Core

Message Extensions

Service Extensions

Meet LTI Advantage

LTI 1.3 Core



Deep Linking

Names and Roles

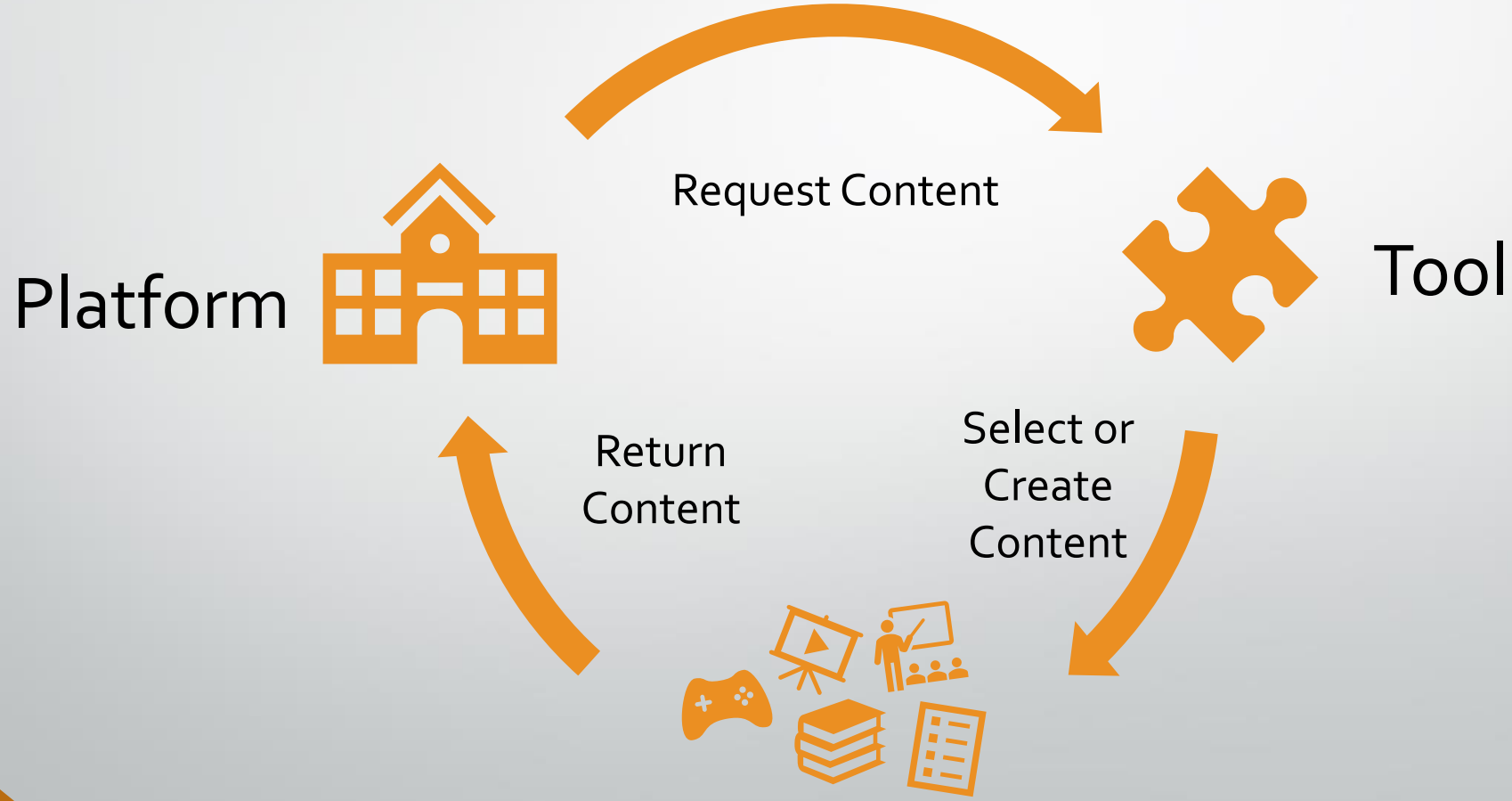
Assignments and Grades

Meet LTI Advantage

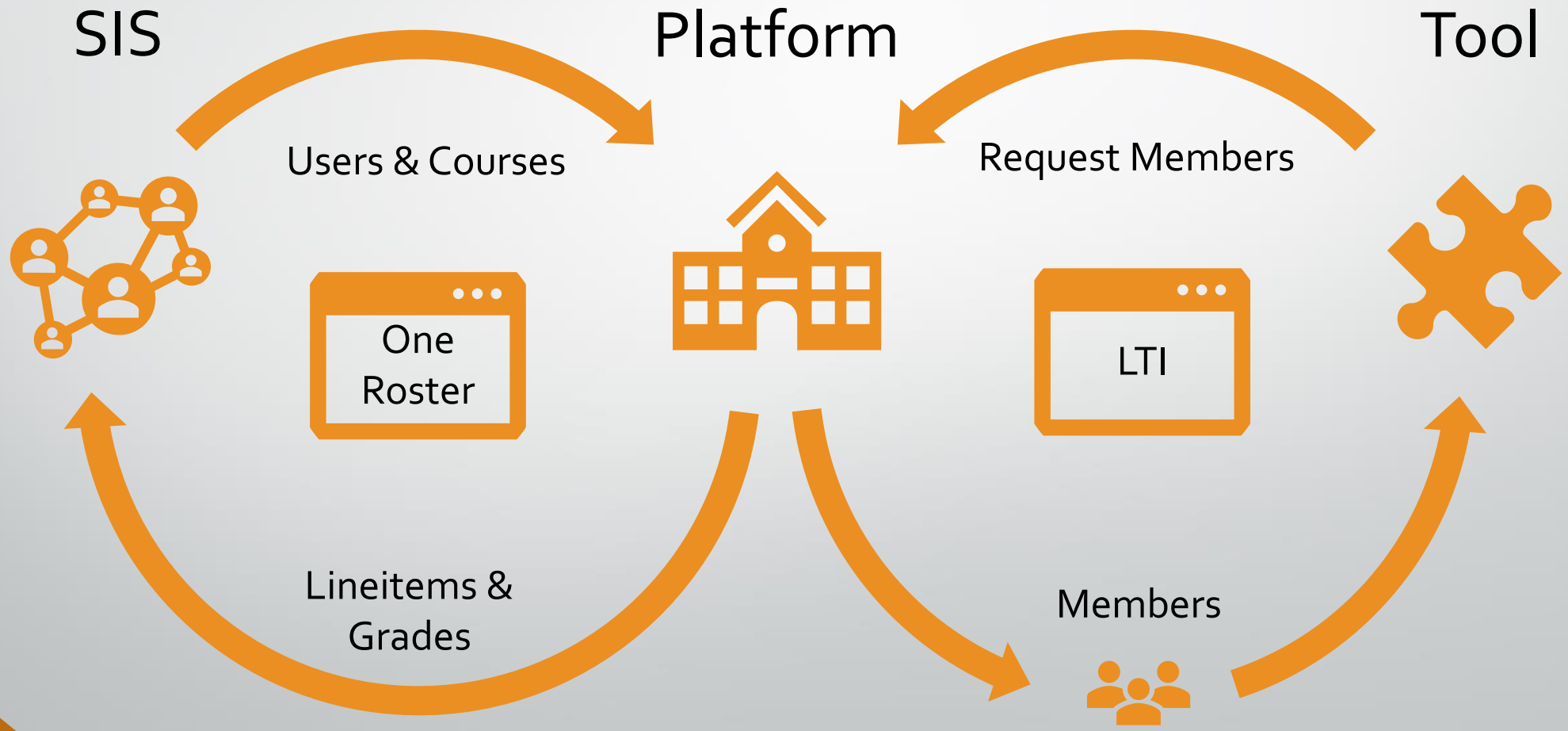
LTI Specs and Extensions

LTI Advantage

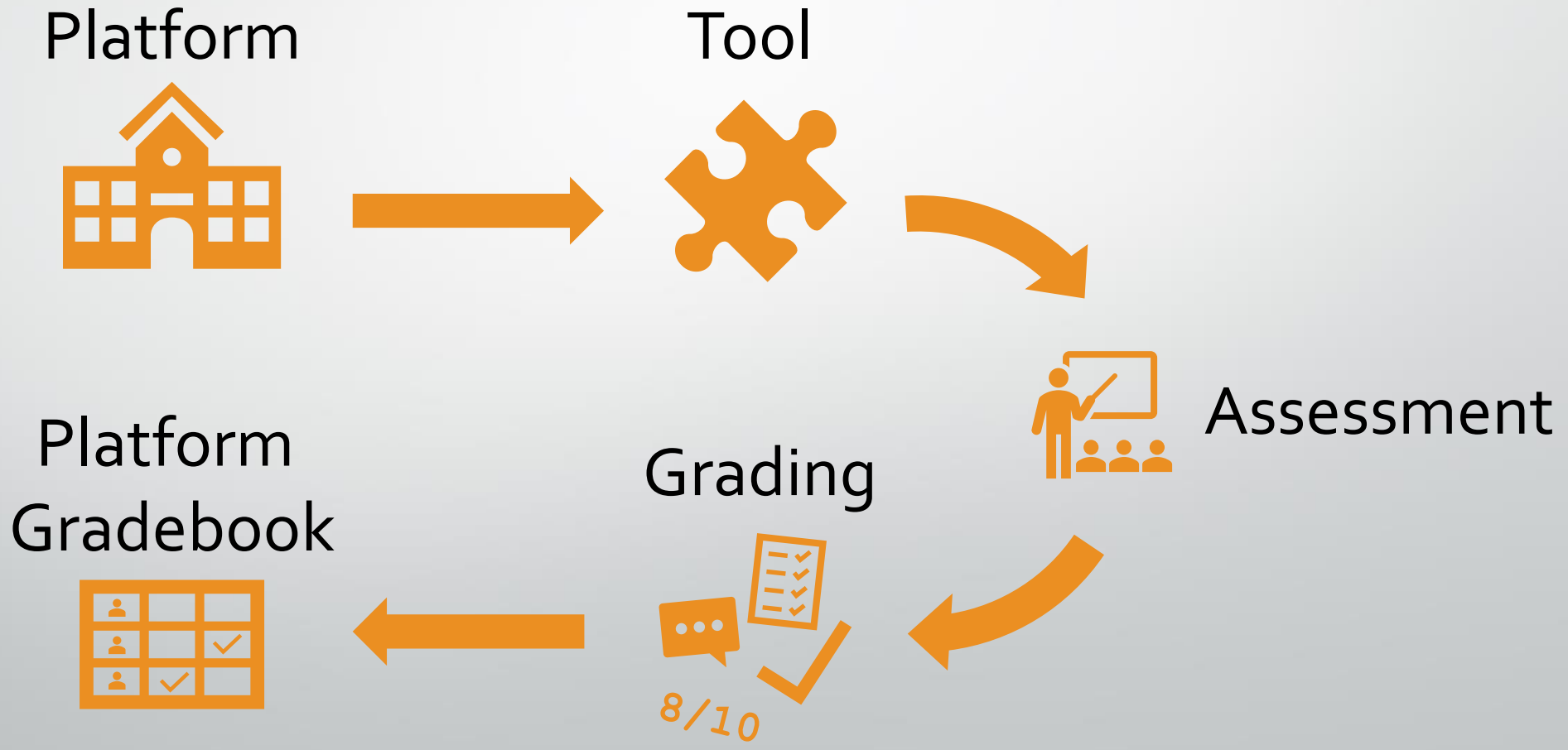
Deep Linking



Names and Roles



Assignments and Grades



What Next?

Course Groups

Submission Review

Caliper Connect

Handy Resources

https://www.imsglobal.org/activity/learning-tools-interoperability

https://github.com/IMSGlobal/ltibootcamp

Take a Break

Coming Up

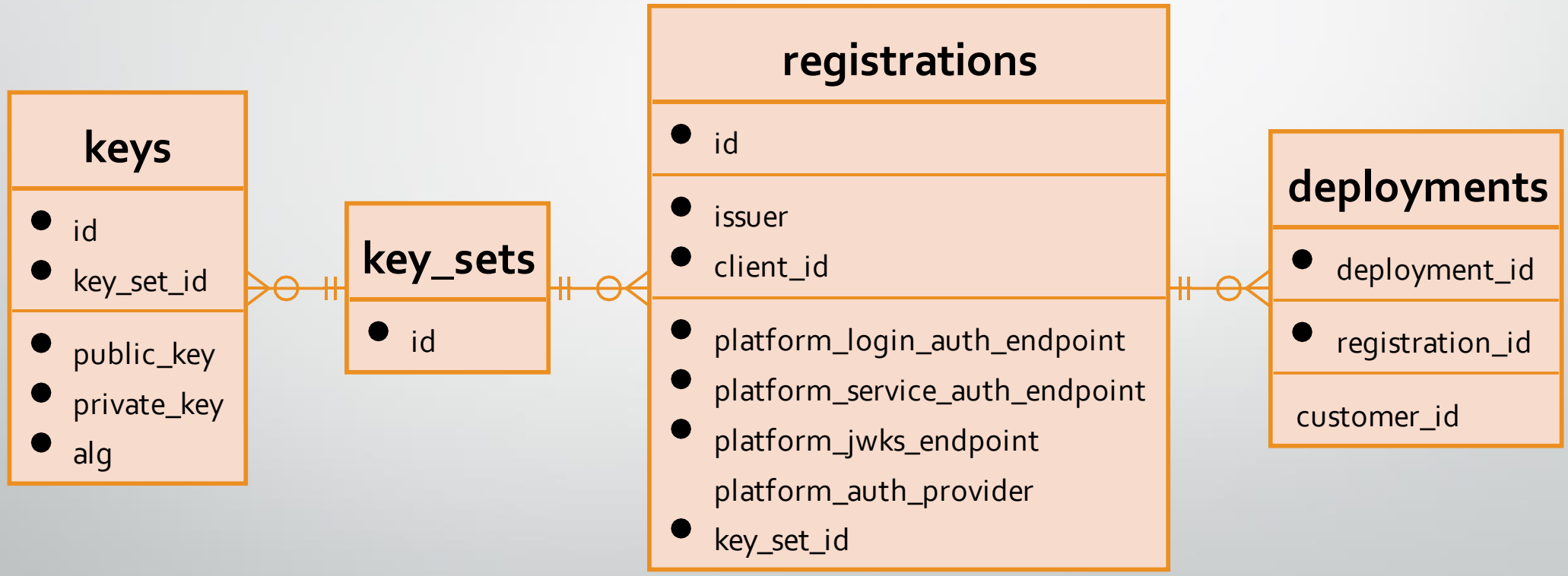
Where should you start?

Doing an LTI message launches

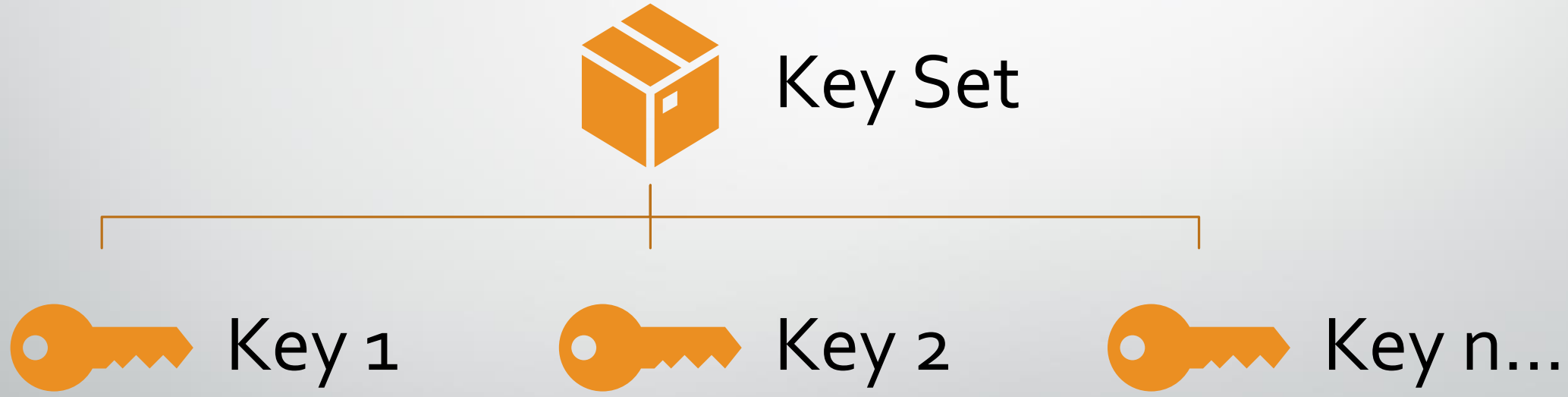
Calling an LTI service

Avoiding “gotchas”

Where to start? – The tool data model



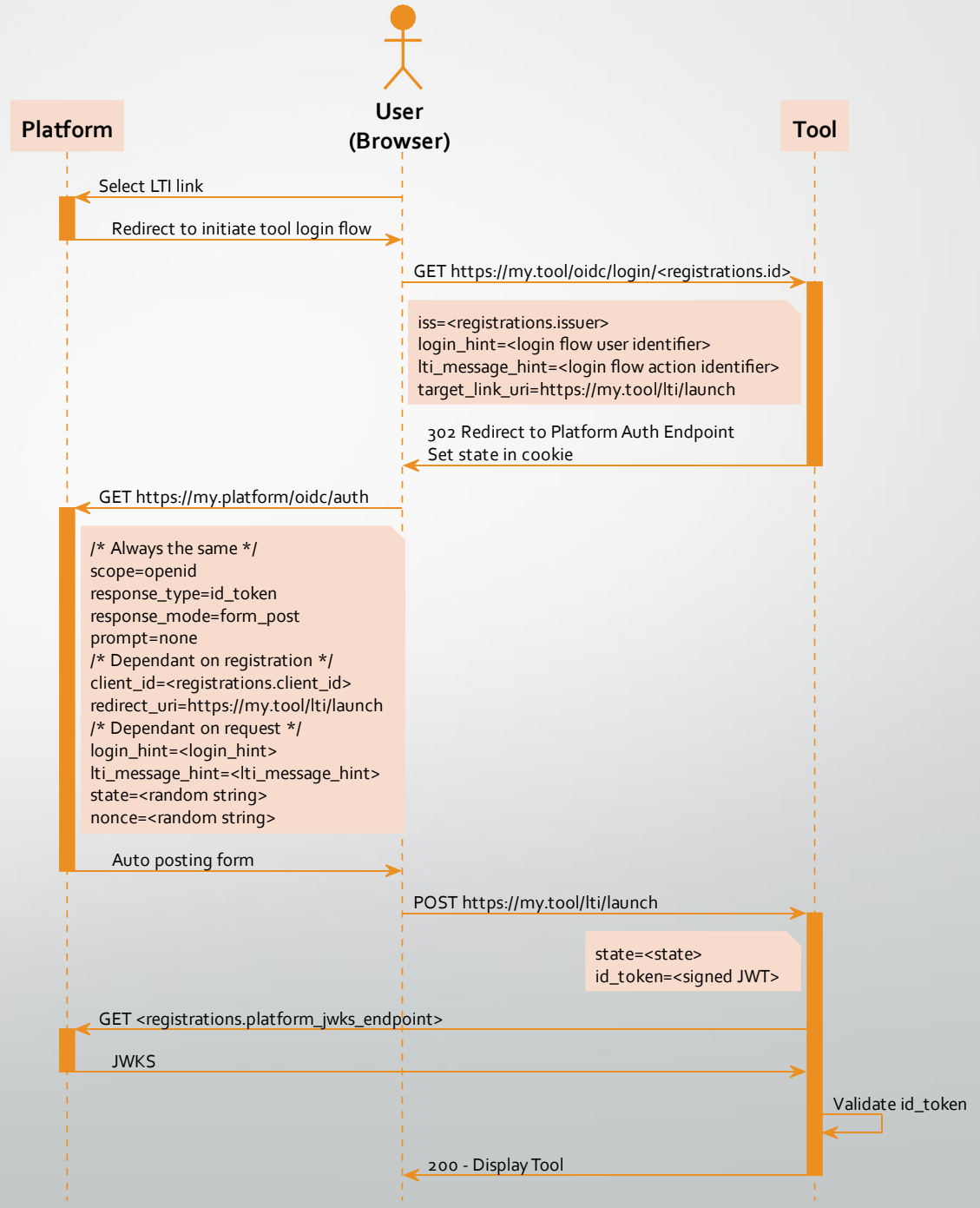
The Keys to the Kingdom



The Keys to the Kingdom
Sending a Message

The Keys to the Kingdom
Receiving a Message

Doing an LTI Message Launch



Message Launch id_token JWT

Header	Message	Signature
• KID (key id) which is used to identify which key in the key set was used to sign a request • ALG which is the algorithm used to sign the request (RS256 must be supported)	• ISS is the issuer of the request (registrations.issuer) • AUD is the audience, which contains the client_id (registrations.client_id). Note: this can be a string or an array of one • deployment_id (deployments.id) is used to identify which deployment of a tool is being used. This can be linked to a customer for the purposes of billing.	• Signature of the JWT header and message. This is signed with the platform's private key and verified with the public key • The public key can be found at the JWKS URL used in the registration (registrations. platform_jwks_endpoint) that matches the KID in the header

LTI Message Claims

iss

- The issuer of the request (registrations.issuer)

aud

- The client id represents the platform's id of the registration (registrations.client_id)

sub

- The user id (Subject) of the user making the request (note this is unique within the issuer)

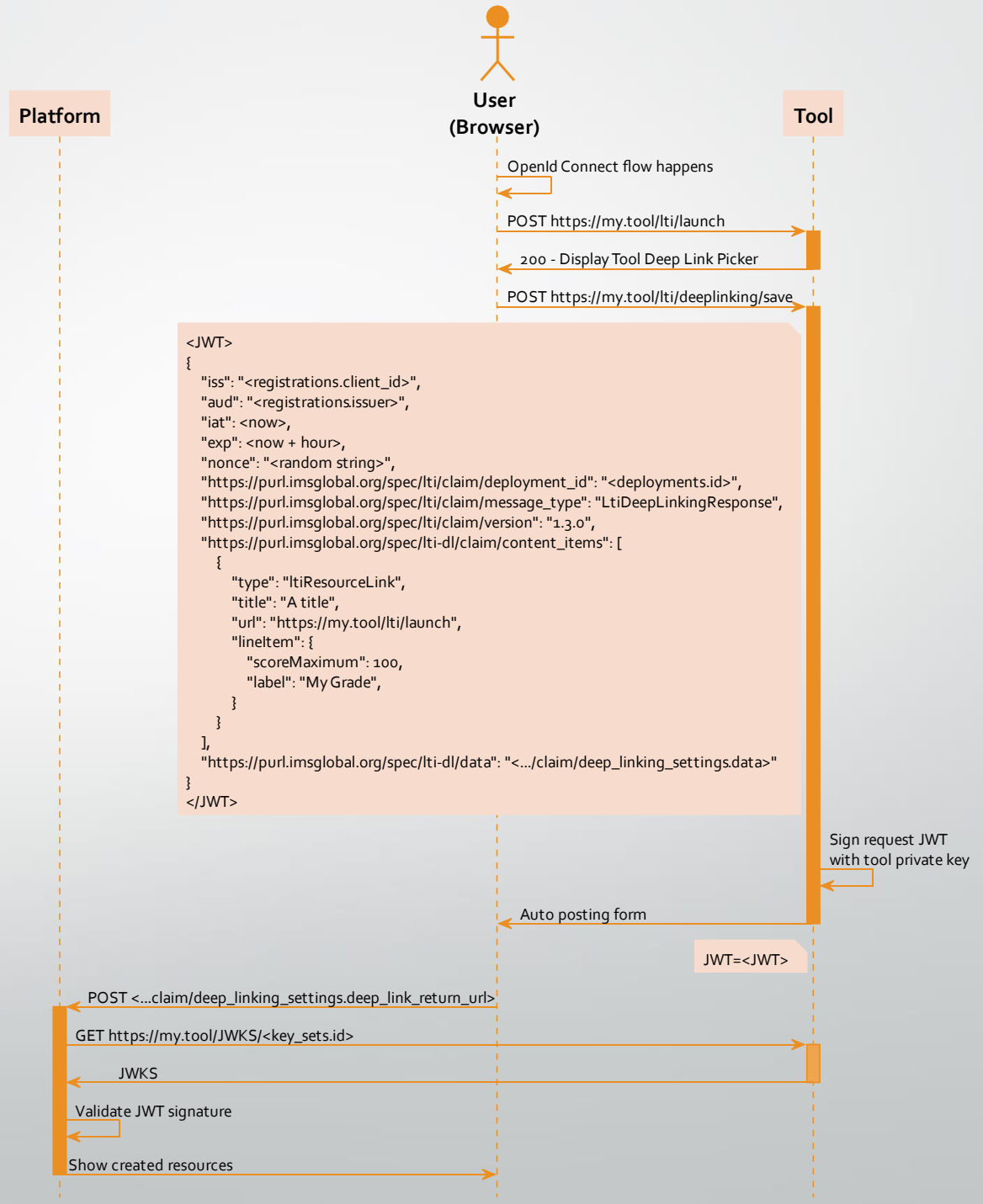
https://purl.imsglobal.org/spec/lti/claim/deployment_id

- The deployment id used to identify an install of a tool onto a platform. This is usually linked to a customer account on the tool side to identify if a customer has access to the tool. Best practice is to prompt a user, if the deployment id isn't recognised, for a login (as an authorized user) to link to an account.

https://purl.imsglobal.org/spec/lti/claim/message_type

- The type of LTI message launch being done. The message type is always present and identifies what other data is available in the message. LTI 1.3 Core includes the LtiResourceLinkRequest message type.

Deep Into Deep Linking Messages



Calling Service Getting a Token

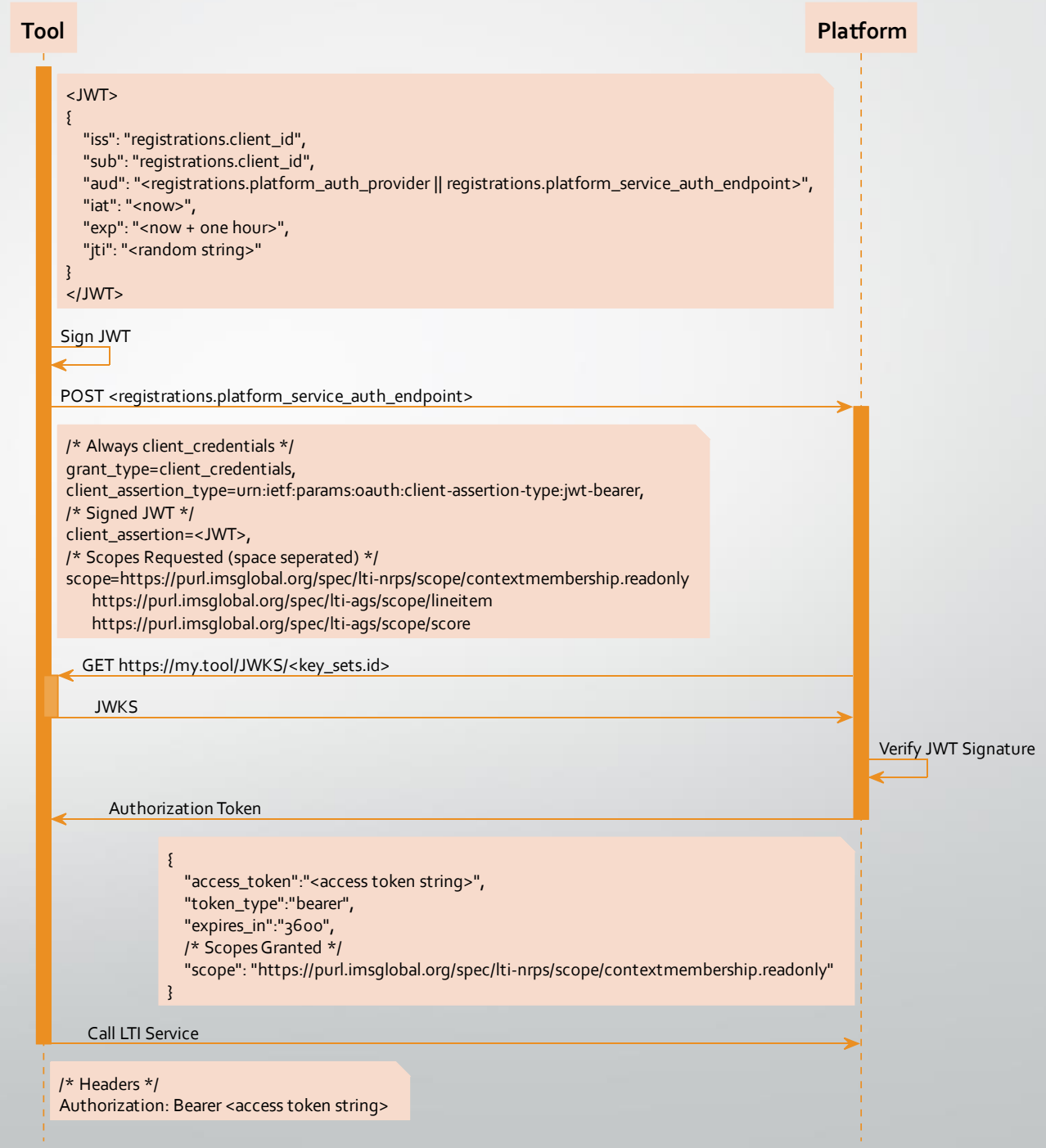
Tool wants to call service

Tool requests a token from platform

Platform mints short lived access token

Tool uses access token to call services

Calling Services
The Gritty Details



Some “gotchas”

- In an id_token, the “aud” can be either an array or a string.
- Launch URLs are pre-registered so cannot change per-launch. For backwards compatibility, use the target link URI in the https://www.imsglobal.org/spec/lti/v1p3/#target-link-uri claim.
- There are three ways a tool’s public key can be shared.
  - Recommended – A tool provides a JWKS URL with their public keys.
  - A tool provides a single public key directly to the platform.
  - A platform generates both the public and private keys for the tool and remembers the public key.
- In a deep linking launch the “data” value given by the platform must be returned unchanged.
- When requesting an access token, the “aud” must be the platform auth provider if given. If no platform auth provider is given the auth token endpoint must be used instead.
- The OpenID Connect login request can be a get or a post.
- A registration can have many deployments, never assume there is only one.

Got an itching for more?

https://www.imsglobal.org/activity/learning-tools-interoperability

https://github.com/IMSGlobal/ltibootcamp